



US005973687A

**United States Patent** [19]

Foley et al.

[11] **Patent Number:** **5,973,687**[45] **Date of Patent:** **Oct. 26, 1999**

[54] **GRAPHICAL DISTRIBUTED MAKE TOOL METHODS APPARATUS AND COMPUTER PROGRAM PRODUCTS**

[75] Inventors: **Jill Foley**, San Jose; **Sunita Ketineni**, Fremont, both of Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Palo Alto, Calif.

[21] Appl. No.: **08/769,620**

[22] Filed: **Dec. 18, 1996**

[51] Int. Cl.<sup>6</sup> ..... **G06F 3/00; G06F 9/44**

[52] U.S. Cl. .... **345/334; 345/349; 345/966; 395/701**

[58] Field of Search ..... **345/349, 966, 345/970, 334; 395/670, 672, 701, 704; 709/100, 102**

[56] **References Cited**

**U.S. PATENT DOCUMENTS**

5,581,797 12/1996 Baker et al. .... 345/326

5,604,908 2/1997 Morton ..... 395/705  
 5,649,085 7/1997 Lehr ..... 345/440  
 5,680,530 10/1997 Selfridge et al. .... 345/440  
 5,731,997 3/1998 Manson et al. .... 364/559  
 5,845,125 12/1998 Nishimura et al. .... 395/704

**OTHER PUBLICATIONS**

Bruce Boardman, "The Next Corporate IS Frontier", Network computing, n718, PG50, pp. 1, 4, and 5, 1996.

*Primary Examiner*—Raymond J. Bayerl

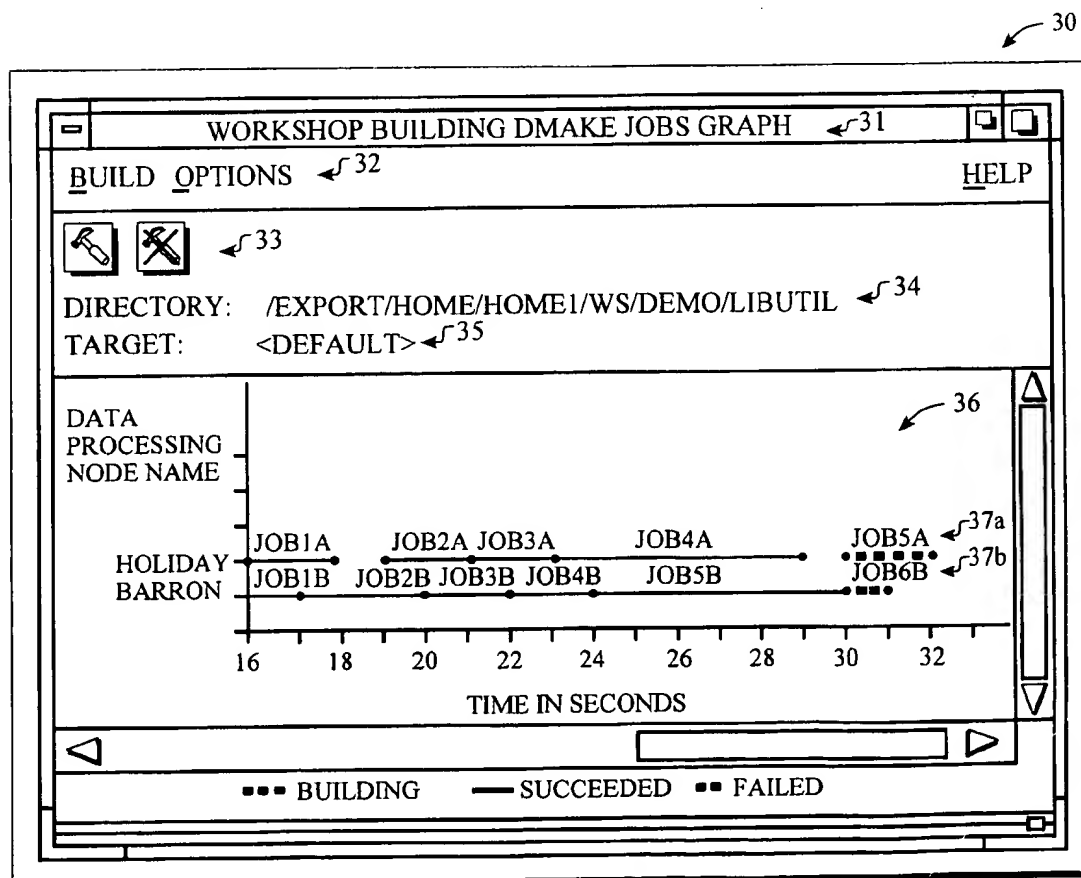
*Assistant Examiner*—X. L. Bautista

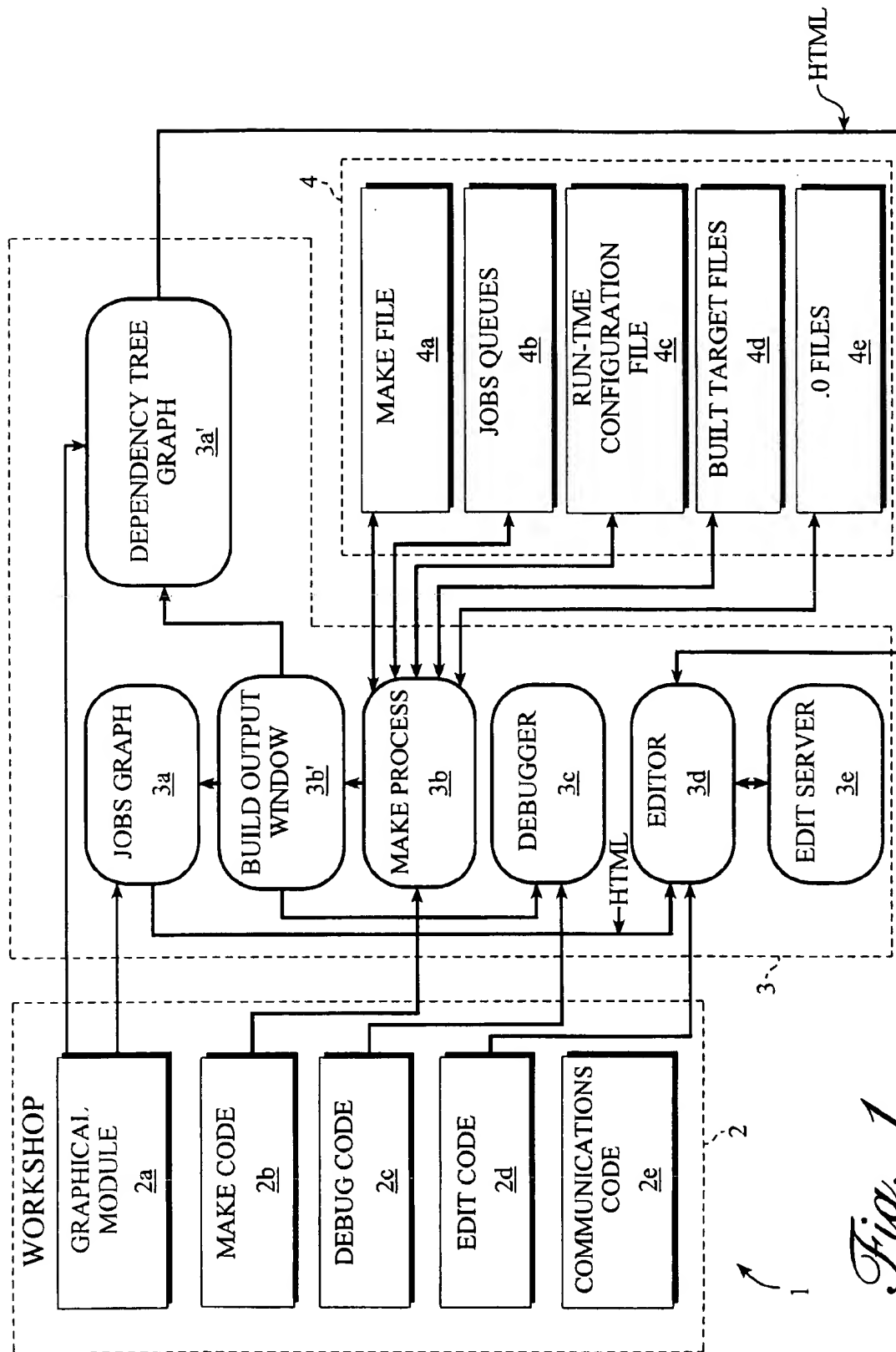
*Attorney, Agent, or Firm*—Sabath & Truong; Robert P. Sabath, Esq

[57] **ABSTRACT**

A make system includes an overlay make tool for graphical presentation of user-friendly data regarding build operations updating multi-file software architecture. The make system includes a make program building files into executable programs, and a make tool which updates files requiring updating according to an update method relying upon a dependency tree and date stamp information.

**17 Claims, 5 Drawing Sheets**





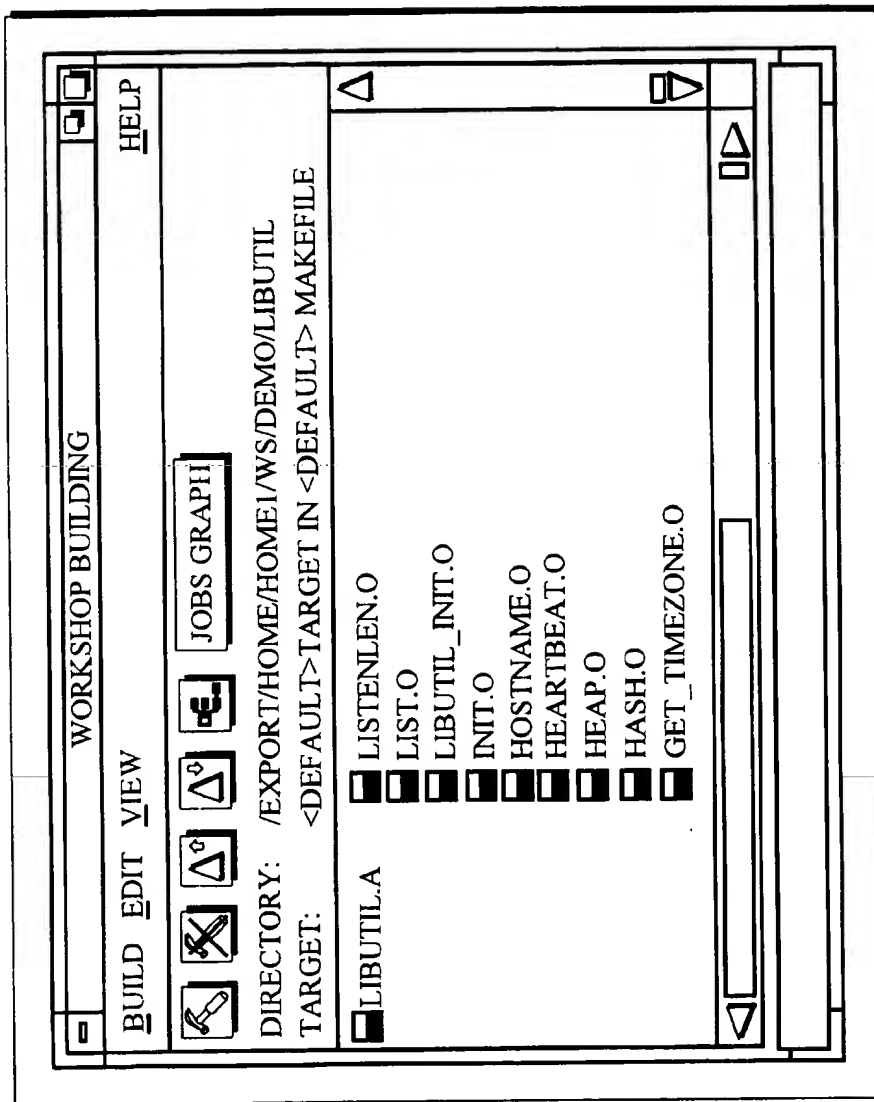
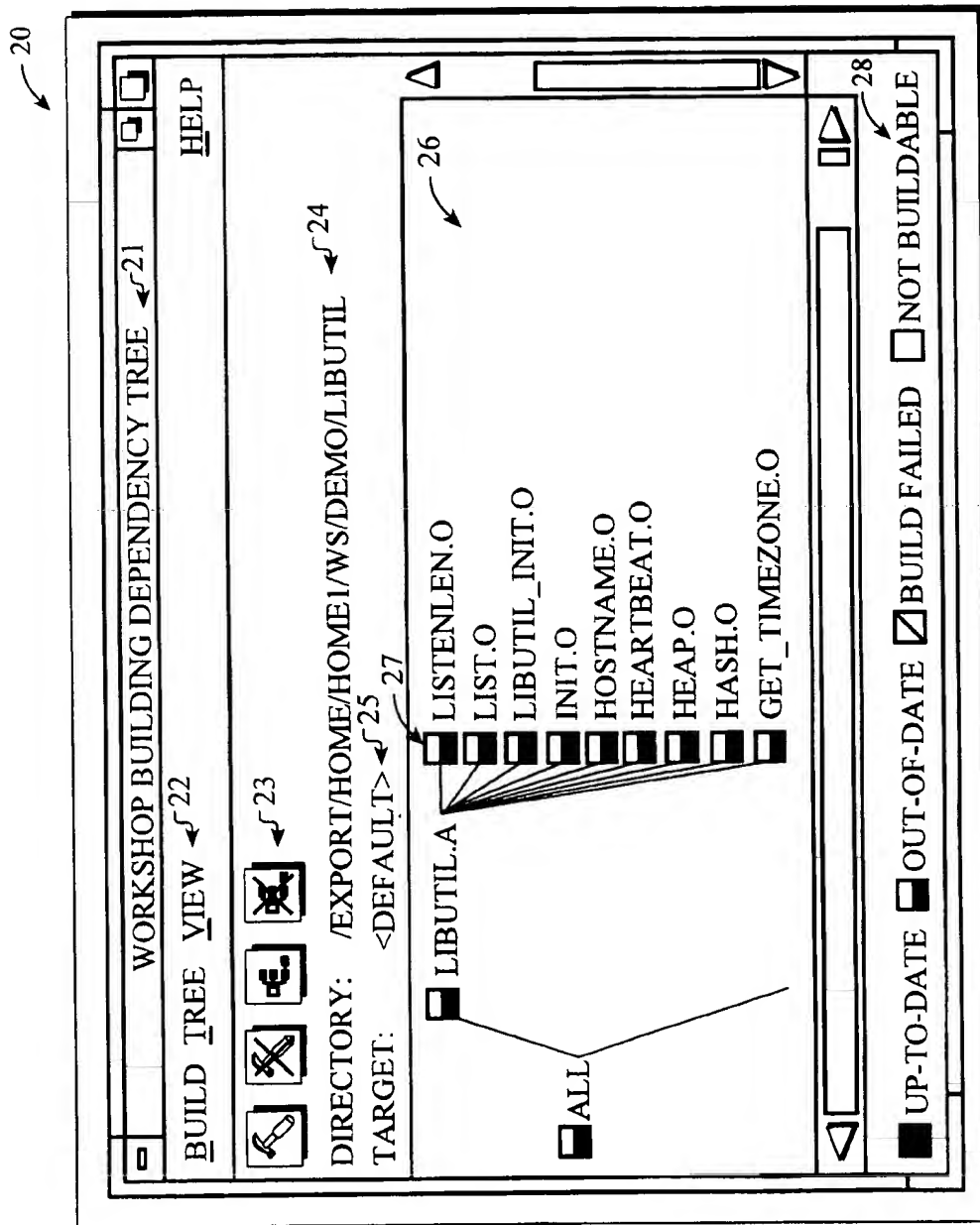
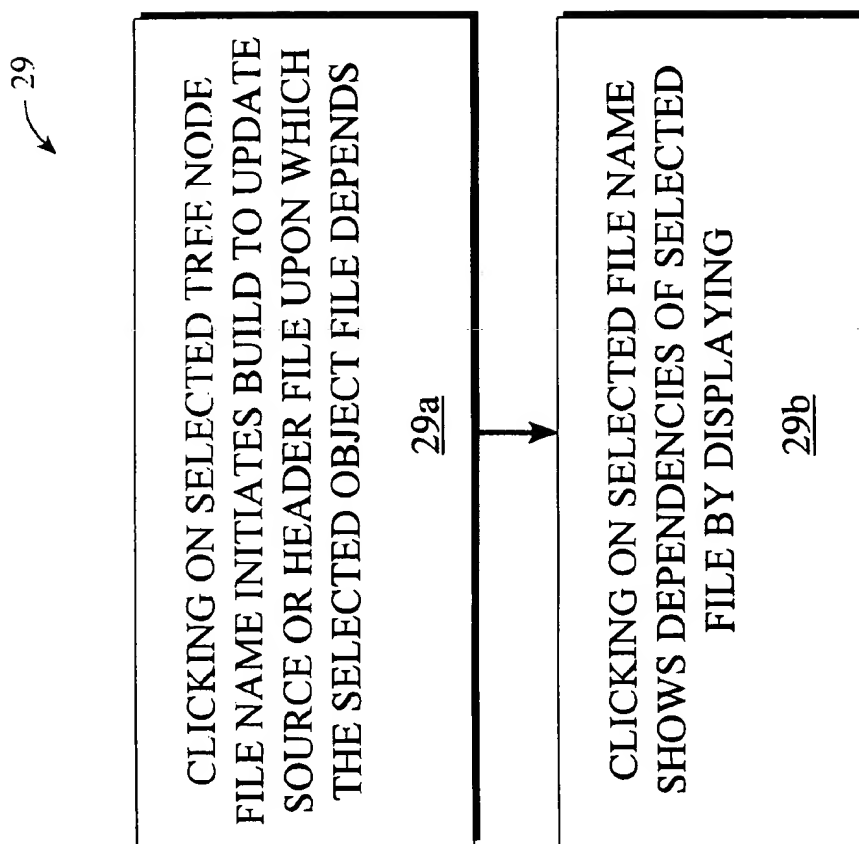
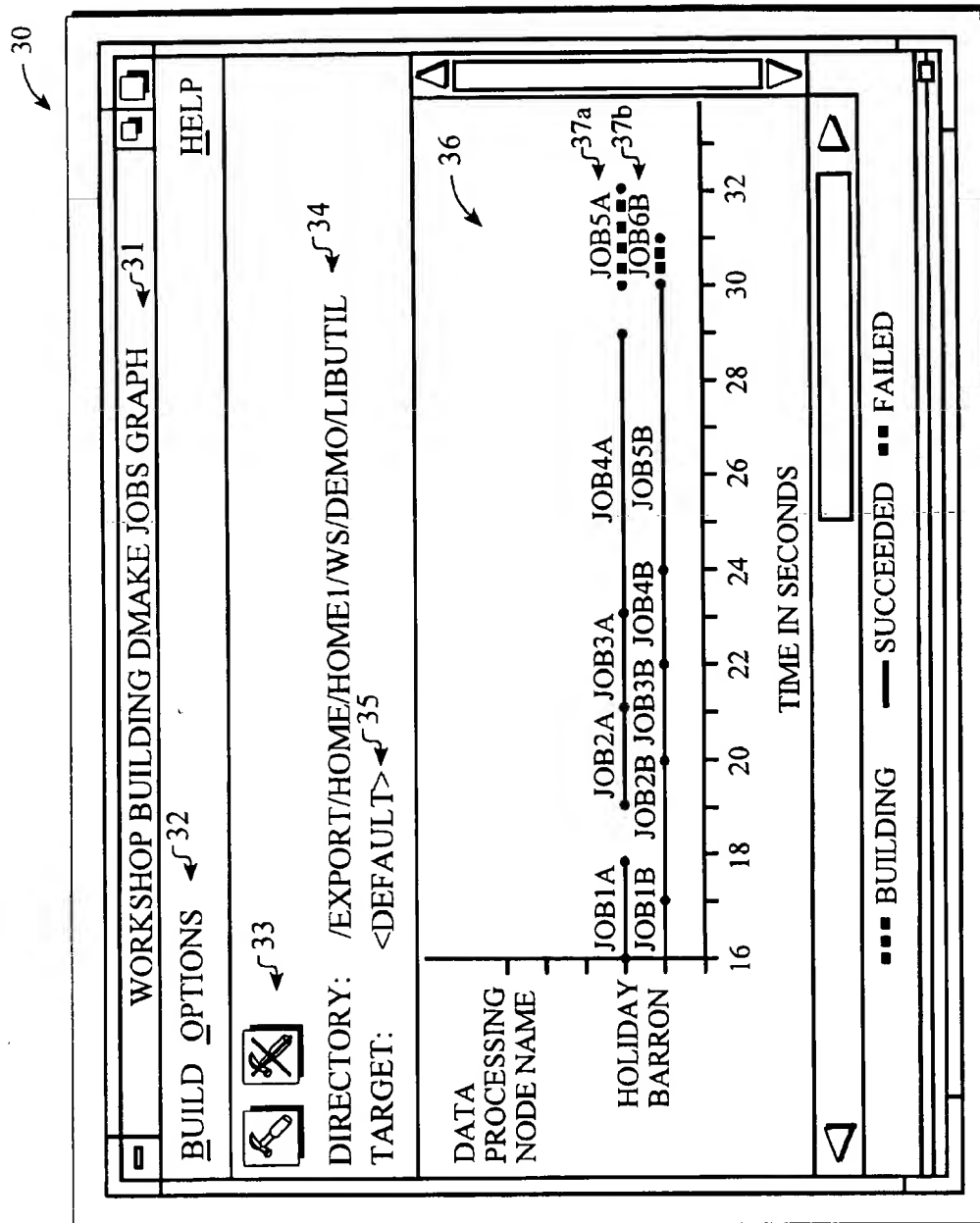


Fig. 2A  
(Prior Art)

*Fig. 2B*

*Fig. 26*

*Fig. 3*

# GRAPHICAL DISTRIBUTED MAKE TOOL METHODS APPARATUS AND COMPUTER PROGRAM PRODUCTS

## CROSS-REFERENCE TO RELATED APPLICATION

This application is related to first and second patent applications filed on even date herewith on "Distributed Make Methods, Apparatus, and Computer Program Products", U.S. patent application Ser. No. 08/920,934, filed Dec. 18, 1996, by Mitchell Nguyen and Wilmer Walton; and "Job Distribution and Local Balancing Methods, Apparatus, and Computer Program Products," U.S. patent application Ser. No. 08/922,103, filed Dec. 18, 1996, by Mitchell Nguyen. The related applications are hereby expressly incorporated herein by reference in their entirety.

## COPYRIGHTS IN PATENT MATERIALS

Portions of this Patent document contain material subject to copyright protection. The copyright owner has no objection to facsimile reproduction of the Patent document after grant as it appears in the U.S. Patent and Trademark Office files and records, but otherwise reserves all rights relating thereto.

## FIELD OF THE INVENTION

This invention relates to methods, apparatus, and computer program products relating to software utilities for modifying and reconstituting software file elements and modules into operable form.

## BACKGROUND OF THE INVENTION

The UNIX environment includes a bundled "make" utility which runs in the UNIX shell, to enable software having multiple source files to be rebuilt into executable or other final form during maintenance. Software maintenance includes updating and modifying software files and then rebuilding files and code dependent upon the changed files. Software being maintained typically includes multiple independent software modules (e.g., .o's). According to the bundled UNIX "make" utility, rebuilding or "build" is accomplished serially by recompiling one source module at a time and then relinking the resultant object files into an executable software product which can be shipped to the customer. According to the prior current source files can be recompiled in parallel at a selected local data processing machine. The cross-referenced applications incorporated herein provide background and details of a current implementation of make. In particular, the make utility identifies source files to be used in rebuilding, compiles the source files serially or in parallel at a single processing node with a suitable compiler, and links the resultant object files into a new rebuilt executable program or other file. A C++ compiler is used to recompile modified .cc object files. A C compiler is used to recompile modified .c object files.

The make utility is useful because it processes many make jobs during build processing which result in creation of an updated resultant software entity which incorporates changes made in precursor code elements used to construct the final entity. The complexity and the workload of the make utility reduces speed of performance. Accordingly, it would be desirable to develop make methods, apparatus and computer program which speed the build process and enables more speedy and effective software maintenance and file modification operations. Furthermore, the software

programmer performing build jobs on multiple files or code elements has little oversight over the status of individual build jobs are completed or interrupted due to error nor on what data processing node a particular job is performed. Additionally, a technical difficulty exists ineffective resolution of errors occurring during make or build jobs. Error diagnosis is thus currently difficult and cumbersome.

## SUMMARY OF THE INVENTION

According to the present invention, the progress of make jobs is presented for visual observation on a graphical user display (GUI) by representing the make jobs performed and executed on one or more data processing nodes as time-expandable job presentation elements on a predetermined multidimensional presentation region viewable on the GUI. Each job is represented by a time-expandable job presentation element arranged at an appropriate position within the predetermined presentation region. According to one embodiment of the present invention, the presentation screen is a two dimensional space having x and y coordinates in which the x-coordinate is arranged in a horizontal alignment and the y-coordinate is vertical and perpendicular to the x-coordinate. According to one embodiment, the job presentation elements are elongated along a coordinate axis which represents time, and the magnitude of each elongated job presentation element represents the current time a job has run during execution up to the present moment. When the job has been completed or when it terminates due to an error condition, the length or magnitude of the particular job presentation element is fixed. According to one embodiment, a job presentation element is provided with a selected first color to indicate that the associated job is still executing. Upon completion of the job, the element changes to a selected second color. If the job terminates due to an error condition, the element converts from being represented in the first color into a selected third color. According to the present invention, job presentation elements relating to jobs processed on each data processing node are associated with each other by adjacent sequential positioning. Similarly, job presentation elements relating to jobs processed on different data processing nodes are separated from each other by a predetermined amount of offset within a common predetermined presentation region. According to one embodiment of the present invention, the offset between the position of job presentation elements relating to jobs processed on different data processing nodes is in a horizontal direction. According to one embodiment, job presentation elements relating to next in time order jobs performed on the same data processing node are dovetailed in adjacent positions.

According to the present invention, each job presentation element is a status element for representing the status of an associated job as a job which is currently being processed, as a job which has already been processed and completed, and as a job which has been processed but which has been terminated due to experience of a processing error. Each of these job statuses is associated with a mode or color, according to one embodiment of the present invention. When the status of a job changes from executing to completed or from executing to terminated due to error, then the mode or color changes from one selected state to another. Further according to the present invention, a computer program product graphically presents the performance of make jobs with a mechanism including code devices configured to execute make jobs and to present status information regarding to several make jobs in relationship to each other, both in time and applicable data process node.

According to one embodiment of the present invention, UNIX build operations are displayed graphically in con-

junction with text output on a graphical user interface (GUI). Further according to the present invention, a graphical UNIX make tool displays file dependencies in tree format and includes a line graph capability to display progress in completing make and build operations. According to one embodiment of the present invention, graphical make is accomplished with line graphing as an overlay for the UNIX make utility. According to the present invention, clicking at a tree node shown on a GUI permits selected file building, as well as file state and currency (i.e., in or out of date) determination. Even further according to the present invention, make tool provides line graph information expressing jobs in progress and indicating which machines are building particular files.

### BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a diagram of a file workshop system according to the present invention;

FIG. 2a is a screen print of a presentation window display of selected file dependencies according to the prior art;

FIG. 2b is a screen print of a tree status graph of files subject to make operation, in accordance with the present invention;

FIG. 2c is a flow chart of a tree graph presentation method according to the present invention, including clicking on selected files to show dependencies; and

FIG. 3 is a screen print of line graphs of files subject to make operation, in accordance with the present invention.

### DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT OF THE PRESENT INVENTION

FIG. 1 is a diagram of a file workshop system 1 according to the present invention. File workshop system 1 includes a plurality of software code modules 2 including 1a integrated form a graphical module 2a, a make code module 2b, a debug code module 2c, an edit code module 2d, and a communications code module 2e. By running software code of software code module 2 according to a method of the present invention, workshop processes 3 are invoked and run as will be discussed in greater detail below. Workshop processes 3 include jobs graph 3a, file dependency tree graph process 3a', build output window process 3b', make process 3b, debugger process 3c, editor process 3d, and edit server daemon process 3e. Workshop processes 3' create and run with data structure and file entities 4 according to the present invention. Data structure and file entities 4 include without limitation a make file 4a, job queues 4b, a runtime configuration file 4a, built target files 4d, and object or o files 4e. New software is created, producing new target source file from a plurality of files including but not limited to object files, i.e., ".o's". The new software which has been built, is then debugged with a user directed debugger process 3c. Debugger process 3c is performed in accordance with instructions provided in debug code module 2c, which may comprise well-known conventional debugging software routines. Debugging identifies "bugs" or errors in the newly created software target files which have been built. The target files built are constructed from subordinate object files which have been compiled and linked into executable or other (e.g., text) file form. To edit or maintain the object files upon which the target software depends, editor process 3c is invoked. Editor process 3d communicates with edit server daemon process 3e to make modifications in .o files to correct errors identified in debugging. Make process 3b uses a make file 4a to identify file dependencies among files which need to be processed to produce a revised built target

file 4d. Make process 3b further processes make jobs taken from job queues 4b created in accordance with the dependencies specified in make file 4a. According to one embodiment, make process 3b distributes the make jobs in job queues 4b to a predetermined plurality of data processing nodes for processing to build target files 4d. According to another embodiment, parallel make processes can be run on a single data processing node. The results of running make process 3b are provided to build output window process 3b' from which a jobs graph process 3a, dependency tree graph process 3a', and debugger process 3c can be invoked by the user. Build output window process 3b' produces a command line interface and permits entry of information by the user to specify build objects identified by directory, target names, commands to be invoked including selection of command modifiers including but not limited to variables, macros, and selected command line options. According to the present invention, jobs graph process 3a and dependency tree graph process 3a' are each linked through editor process 3d to modifiable files upon which target files depend, such as .o files 4e, by a hypertext markup language (HTML) link.

FIG. 2a is a screen print of a presentation window display of selected file dependencies according to the prior art. In particular, FIG. 2a shows a workshop build window including a tool bar, task icons, a directory indication, a target line, and a text region showing a listing of object files subject to conventional build operation. The title of the build window is "Workshop Building," which refers to a software suite including conventional build functionality. The window presentation, according to this known technology, conveys only a limited amount of information. The window presentation, however, provides no information about currency status, i.e., whether a particular file is up-to-date or out-of-date. Further, no information is provided regarding file status, i.e., whether build failed or whether particular files indicated are not buildable.

FIG. 2b is a screen print of a dependency tree graph 20 of the names of files subject to make operation, in accordance with the present invention. In particular, FIG. 2b shows dependency tree graph 20 including a header 21, a menu 22, task icons 23, a directory indication 24, a target line 25, and a graph region 26 in which a dependency tree of files subject to make operation is presented in terms of their interdependencies. Each file name is shown associated with a rectangular icon 27 according to a legend 28 at the underside of the dependency tree graph window indicative of file currency and status. For example, object files listelem.o-heap.o are shown out-of-date, while object files hash.o-get\_timezone.o are shown to be up-to-date. No files are indicated as build failed or as not buildable. All of these files are dependencies of libutil.a, and all is a dependency of libutil.a. Since some of the files from listelem.o-get\_timezone.o are out-of-date, it follows that libutil.o and all will have an icon indication of their being out-of-date as well. This status is expressly visually apparent from the tree organization shown in the user-friendly presentation in FIG. 2b.

FIG. 2c is a flow chart of a dependency tree presentation method 29 according to the present invention. According to the present invention, clicking 29 on a selected tree node file name initiates build operation according to the present invention, to update the source and header files upon which the selected object file depends. Further, clicking 29b on a selected object file name causes presentation of the names of the source and header files upon which the selected file depends.

FIG. 3 is a screen print of line graphs of files subject to make operation, in accordance with the present invention. In



particular, FIG. 3 shows a jobs graph 30 including a header 31, a menu 32, task icons 33, a directory indication 34, a target line 35, and a two dimensional graph region 36 in which first and second line graphs, respectively 37a and 37b, illustrate build status in a job-by-job basis, as a function of time in seconds. Each of line graphs 37a and 37b comprises a series of job presentation elements or segments which have a limited thickness or width, but each of which has a length corresponding to the processing time consumed in executing or running the job associated with the element. Each element can exhibit a mode or color, according to the status of the job as running, failed, or completed. According to one embodiment, a running job is represented by a solid segment or the color green, as it grows or extends as a function of time. As shown in FIG. 3, make jobs are being performed on first and second data processing nodes, respectively named "holiday" and "barrone." Running on first data processing node "holiday" are jobs named JOB1A-JOB5A. Running on second data processing node "barrone" are jobs named JOB2B-JOB6B. As can be seen, at time 16 JOB1A has begun running and has continued to run for two seconds until it has succeeded in completing the specified task or tasks associated with the job. Then, for one second, data processing node "holiday" ceases processing. Thereafter, at time 19, JOB2A starts and upon its completion at time 21, JOB3A starts and continues processing until time 23, when JOB4A starts and continues processing until time 29. Again, data processing node "holiday" ceases operation, only to start up again at time 30 with JOB5A which is still building, i.e., in process at the moment the screen shot of FIG. 3 is captured. Parallel jobs named JOB1B-JOB6B have run or have failed as indicated for data processing node "barrone." As shown in FIG. 3, JOB1B has started prior to time 16. Earlier job data is not provided in the present setting of the job graph limited to times from 16 seconds to 33 seconds. As further shown, JOB6B has failed. The earlier jobs handled by "barrone" have succeeded serially without "barrone" halting processing at any time.

By clicking on the job processing element for JOB 6B, according to one embodiment of the present invention, a pull-up screen can be shown which presents a table of error diagnostics which are the reasons for the failure of JOB 6B. Further according to the present invention, selected words in the error diagnosis table are hyperlinked to modifiable code in files which are subject to edit operation in accordance with the present invention. The appendix contains an embodiment of hypertext markup language (HTML) code to permit the linkage to code containing errors which can be corrected by edit operations according to the present invention. The appendix which follows further includes code to implement graphs in accordance with the present invention. The portion of the appendix entitled "mt\_graph.cc" is code for graph generation, and the portion entitled "BuildOutputHTML.cc" is the code for facilitating hyperlink corrections to code needing to be modified to prevent occurrence of error conditions.

What is claimed is:

1. A computer implemented method of graphically presenting the performance of a plurality of make jobs being executed by at least one data processing mode, comprising:
  - presenting status information regarding processing said plurality of make jobs as a function of dependencies between ones of said plurality of make jobs, said status information including at least one of build failure status and non-buildable status;
  - presenting first status information regarding processing a first one of said plurality of make jobs as a first function

- of time, at a selected first position within a predetermined presentation region having at least a single dimension; and
- presenting second status information regarding a second one of said plurality of make jobs as a second function of time, at a second position within said predetermined presentation region.
2. A computer implemented method according to claim 1, further including presenting said first and second status information regarding said first and second ones of the plurality of make jobs adjacently to each other within said presentation region.
3. A computer implemented method according to claim 1, including offsetting said second position from said first position to present said first and second status information with respect to make jobs performed with different ones of said at least one data processing node, at predetermined intervals along a selected dimension of said common presentation region.
4. A computer implemented method according to claim 3, including offsetting the first and second positions of said respective first and second ones of the plurality of make jobs from each other in a horizontal direction.
5. A computer implemented method according to claim 3, including offsetting the first and second positions of said respective first and second ones of the plurality of make jobs in a vertical direction.
6. A computer implemented method according to claim 1, including dovetailing the presentation of said ones of the plurality of make first and second status information when performed on a single data processing node, so as to display a first indication of performance of said first one of the plurality of make jobs at said first position followed by a second indication of performance of said second one of the plurality of make jobs after said first display location.
7. A computer implemented method according to claim 1, including presenting first and second status information regarding job performance with a similar status element type representing jobs being processed, jobs successfully having been processed and completed, and jobs processed by subject to processing error.
8. A computer implemented method according to claim 1, including presenting status information regarding respective jobs being processed, in a selected color to indicate job processing is currently in progress.
9. A computer implemented method according to claim 1, including presenting status information regarding jobs being processed, in a selected color to indicate job processing has successfully been completed.
10. A computer implemented method according to claim 1, including presenting status information regarding jobs being processed, in a selected color to indicate job processing is subject to error.
11. A computer implemented method according to claim 1, including presenting job status information for jobs currently being processed with a time function expanding element.
12. A method according to claim 3, including changing the color of status information to reflect a change in processing status.
13. A computer program product, comprising:
  - a computer storage medium and a computer code mechanism embedded in the computer storage medium for graphically presenting the performance of make jobs, the computer program code mechanism comprising:
    - a first computer code device configured to execute first and second make jobs with at least a single data processing node;

7

a second computer code device configured to present status information regarding said first and second make jobs at respective first and second positions within a predetermined graphical presentation region; and

a third computer code device configured to present status information regarding processing said make jobs as a function of dependency between ones of the make jobs, said status information including at least build failure status and non-buildable status.

14. A computer apparatus comprising at least a single data processing node configured to execute a plurality of make jobs, at least one of said at least a single data processing node further being configured to present first status information about said plurality of make jobs within a multidimensional graphical presentation region receiving real-time information about the job status of said respective jobs, said first status information being presented as a function of time and the identity of the particular one of said at least a single data processing node which performs particular job processing at least one of said, at least a single data processing node further being configured to present second status information regarding processing said make jobs as a function of dependency between ones of the make jobs, said status information including build failure status and non-buildable status.

15. A computer implemented method for presenting make job performance information including:

8

providing a mechanism for executing make jobs;

presenting status information regarding processing said plurality of make jobs as a function of dependencies between ones of said plurality of make jobs, said status information including at least one of build failure status and non-buildable status; and

presenting time and status varying information on each make job executed according to actual make job performance sequence, on a predetermined presentation region.

16. A computer implemented method of responding to a failed make job in a first graphical representation of the progress of make jobs in which each make job is represented by a first hyperlink activable job representation element which has a magnitude corresponding to the time to complete processing or the time to job failure or in a second graphical representation of the progress of make jobs in which each make job is represented by a second hyperlink activable job representation element which has a symbol indicative of a build failure status or a non-buildable status, comprising:

activating a selected hyperlink activable job representation element; and linking to job associated code causing the job failure.

17. The method of claim 16 further comprising displaying said job associated code to allow changes to said code.

\* \* \* \* \*



US006567108B1

(12) **United States Patent**  
Master et al.

(10) Patent No.: **US 6,567,108 B1**  
(45) Date of Patent: **May 20, 2003**

(54) **COMPUTER IMPLEMENTED METHOD OF IMPLEMENTING, MAINTAINING, AND UPDATING A BRAND ARCHITECTURE SYSTEM**

(76) Inventors: **Syndi Beth Master**, 1995 University Ave. Ste. 510, Berkeley, CA (US) 94704; **Timothy J. M. Symons**, 4 Greenwood Common, Berkeley, CA (US) 94708

(\*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/438,739**

(22) Filed: **Nov. 11, 1999**

(51) Int. Cl.<sup>7</sup> ..... **G06F 3/00; G06F 17/60**

(52) U.S. Cl. .... **345/853; 345/765; 345/825; 345/826; 345/962; 705/10; 706/47**

(58) Field of Search ..... **345/331, 339, 345/353, 356, 357, 440, 962, 971, 764, 765, 810, 818, 825, 826, 835, 841, 853, 854; 705/7, 10; 706/46-48, 60, 925**

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,701,400 A \* 12/1997 Amado ..... 706/47 X

5,995,951 A \* 11/1999 Ferguson ..... 706/10

6,029,139 A \* 2/2000 Cunningham et al. .... 705/10

6,078,922 A \* 6/2000 Johnson et al. .... 705/10 X

6,119,101 A \* 9/2000 Peckover ..... 705/10 X

\* cited by examiner

*Primary Examiner*—Raymond J. Bayerl

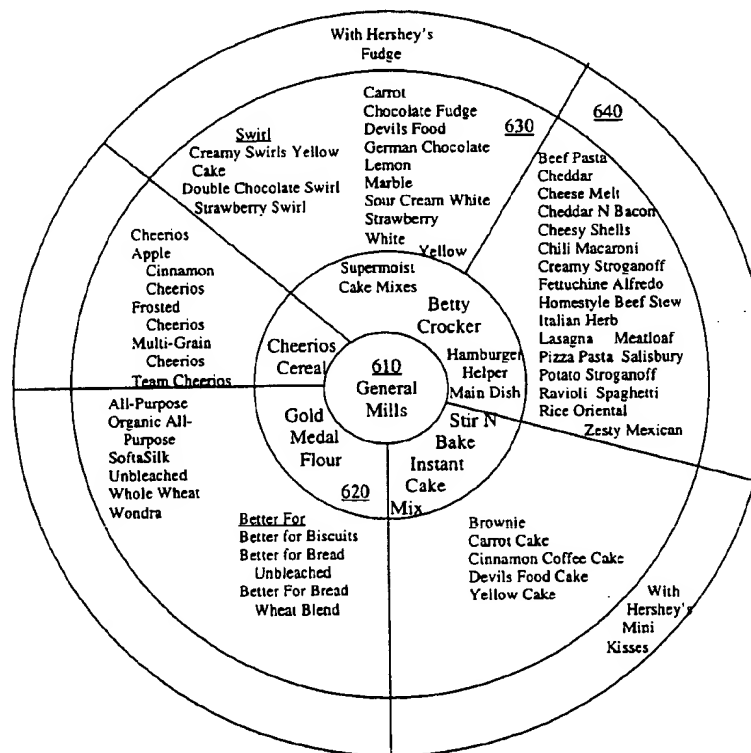
*Assistant Examiner*—X. L. Bautista

(74) *Attorney, Agent, or Firm*—Blakely, Sokoloff, Taylor & Zafman, LLP

(57) **ABSTRACT**

A method of implementing an automated brand architecture system is described. The method comprises the computer implemented steps of displaying a navigable brand-architecture chart. The method further comprises receiving a nomination for a new name including a location in the brand-architecture chart. The method further includes testing the name against rules, to see whether the new name fits criteria. If the new name fits criteria, the brand architecture chart is updated with the new name.

**35 Claims, 10 Drawing Sheets**



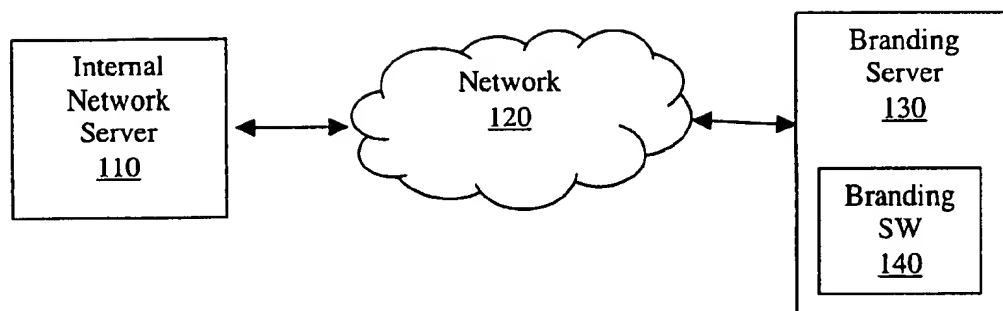
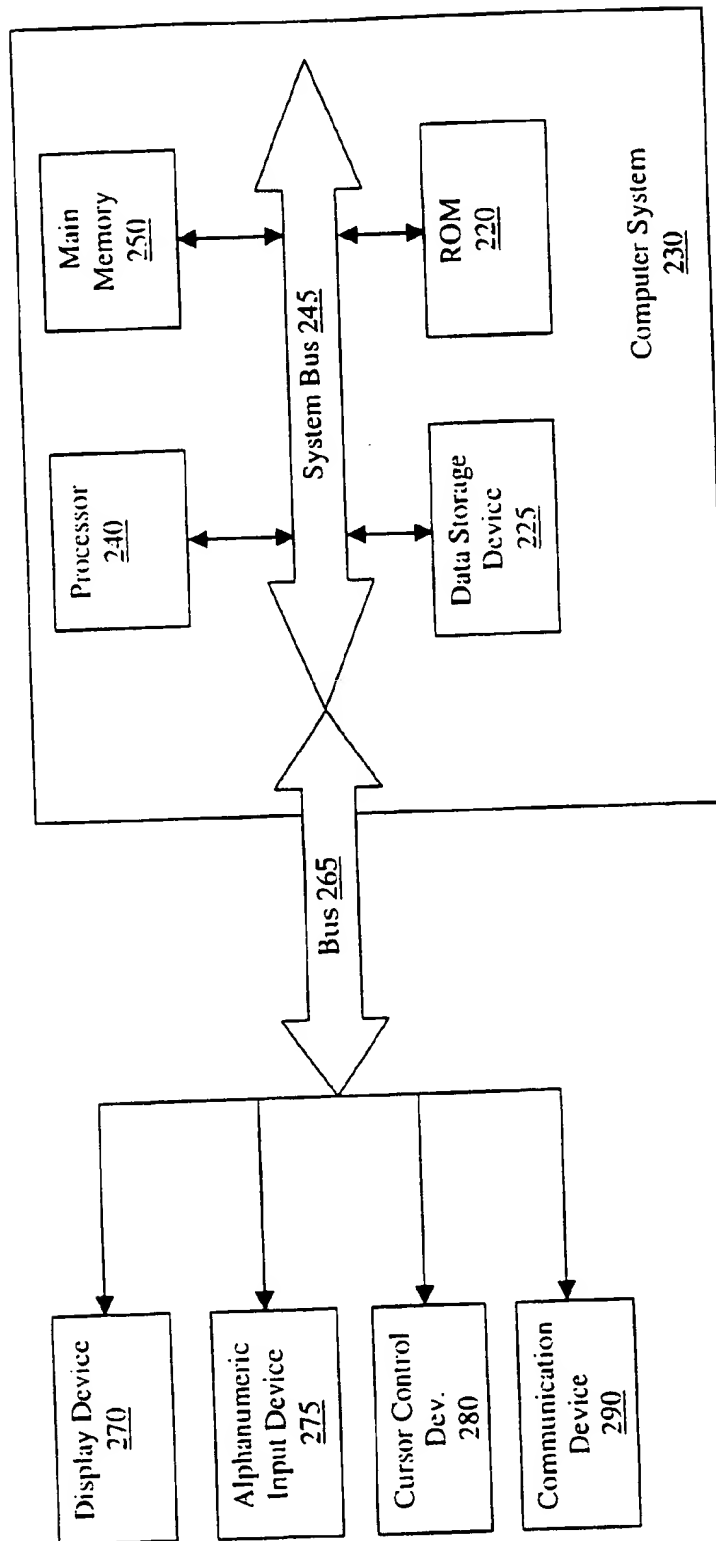


Fig. 1

**Fig. 2**

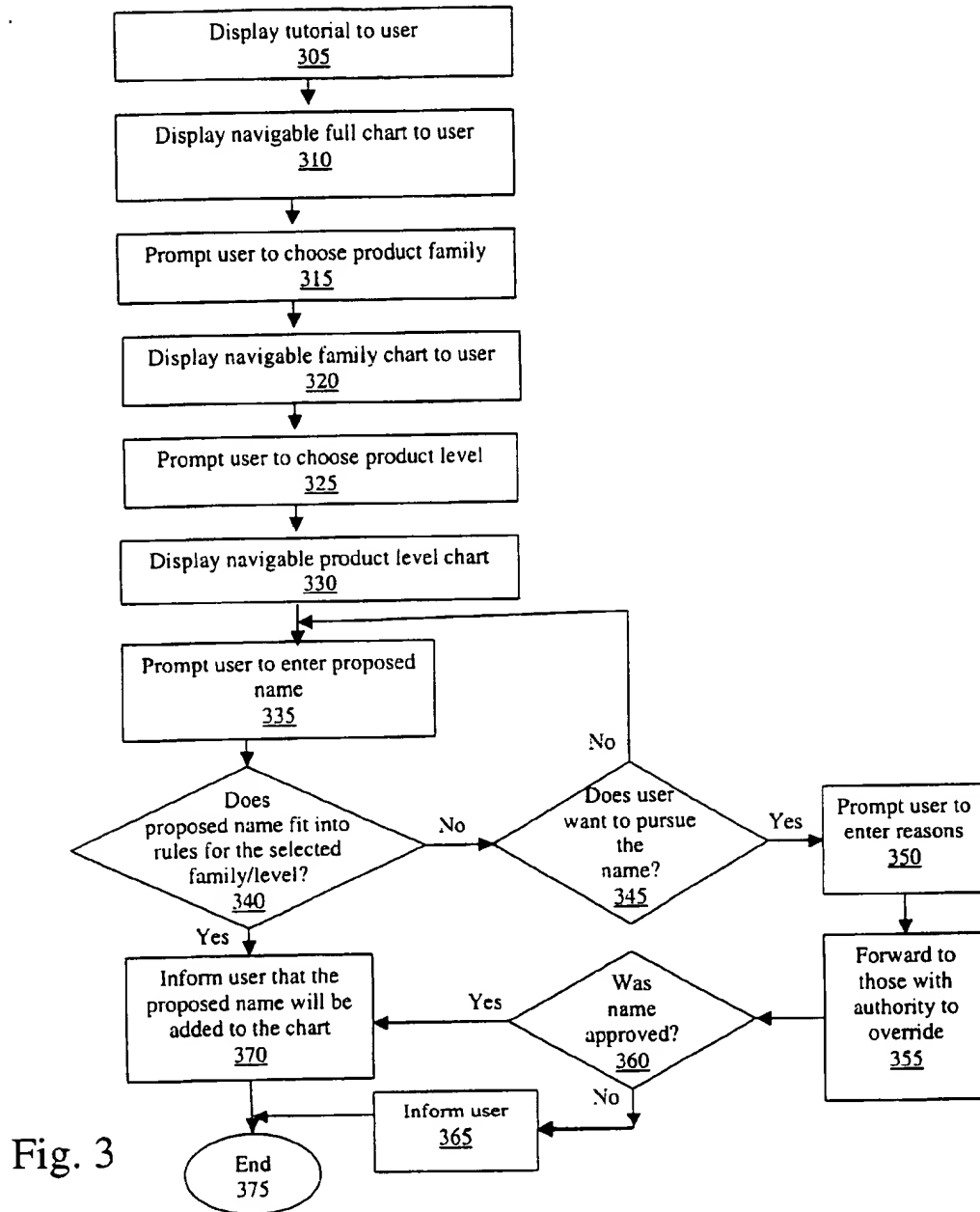


Fig. 3

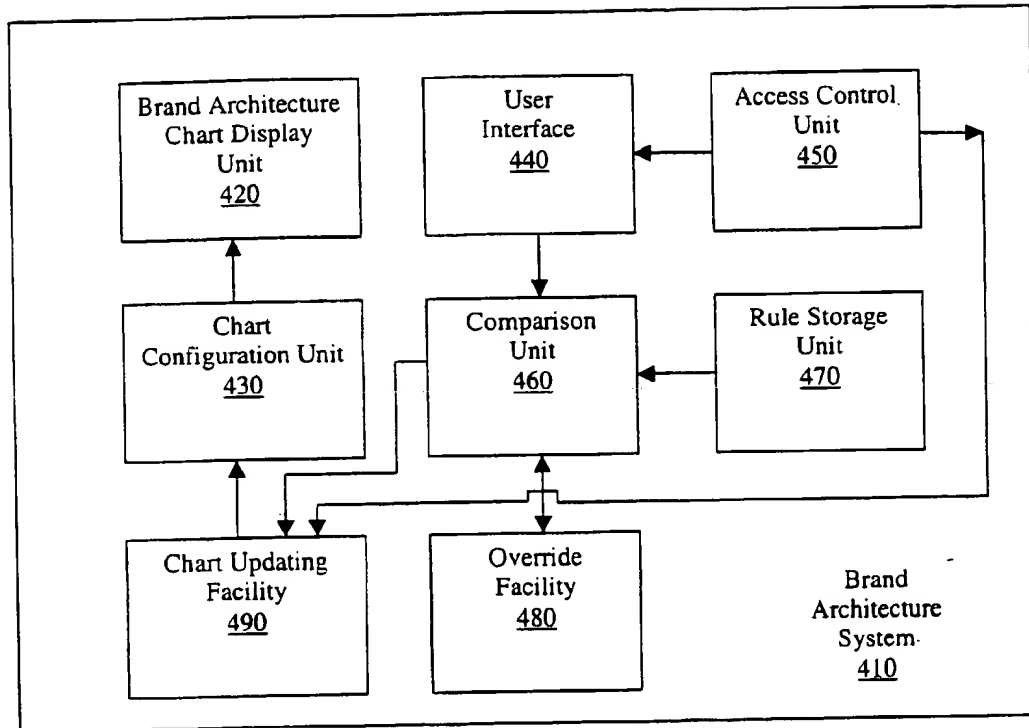


Fig. 4

510 {	Name	
	Title	
	Group	
	Contact Information	
520 ➡	First Level	Brand/Product or Ingredient?
530 ➡	Second Level	Within what Brand/Product/or Ingredient list?
540 ➡	Definition of Product	
550 ➡	Proposed Product Name	
560 ➡	Proposed Support level for product	
570 ➡	Projected income from Product	
580 ➡	Does user request help with naming?	

Fig. 5



GENERAL MILLS					
Corporate Name	Cheerios Cereal	Betty Crocker		Stir N Bake Instant Cake Mix	Gold Medal Flour
Product Family Brands		Super Moist Cake Mix	Hamburger Helper Main Meal Dish		
Products	Cheerios Apple Cinnamon Cheerios Frosted Cheerios Honey Nut Cheerios Multi Grain Cheerios Plus Team Cheerios	Carrot Chocolate Fudge Devils Food German Chocolate Lemon Marble Sour Cream White Strawberry White Yellow  Swirl Creamy Swirls Yellow Cake Double Chocolate Swirl Strawberry Swirl	Beef Pasta Cheddar Cheese Melt Cheddar N Bacon Cheesy Shells Chili Macaroni Creamy Stroganoff Fettuccine Alfredo Homestyle Beef Stew Italian Herb Lasagna Meatloaf Pizza Pasta Potato Stroganoff Ravioli Rice Oriental Salisbury Spaghetti Zesty Mexican	Brownie Carrot Cake Cinnamon Coffee Cake Devils Food Yellow Cake	All-Purpose Organic All Purpose Softasilk Unbleached Whole Wheat Wondra  Better for Better for Biscuits Better for Bread Unbleached Better for Bread Wheat Blend
Special Ingredients		With Hershey's Fudge		With Hershey's Mini Kisses	

FIGURE 6A

GENERAL MILLS				
Corporate Name				
Product Family Brands	Cheerios Cereal	Betty Crocker		Gold Medal Flour
		Super Moist Cake Mix	Stir N Bake Instant Cake Mix	
		Hamburger Helper Main Meal Dish		

FIGURE 6B

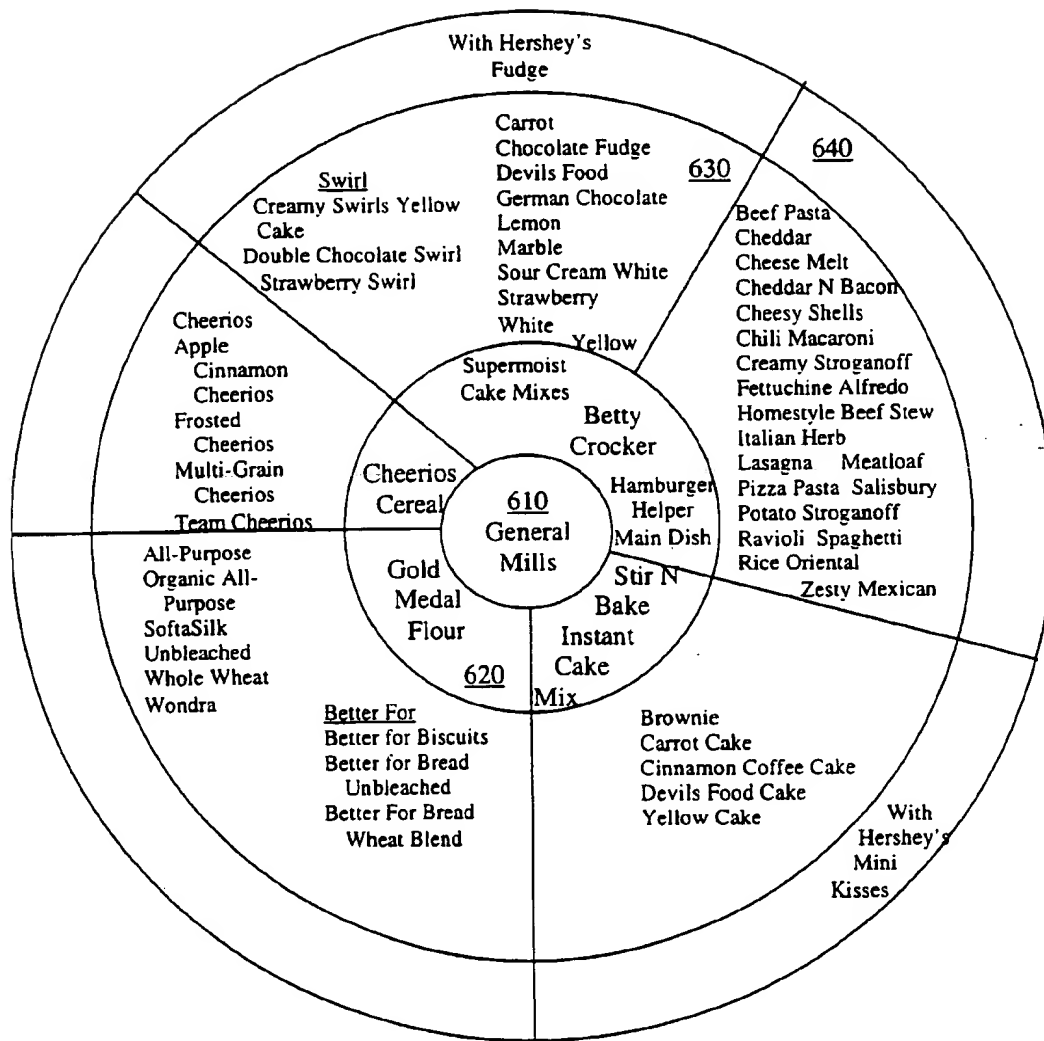
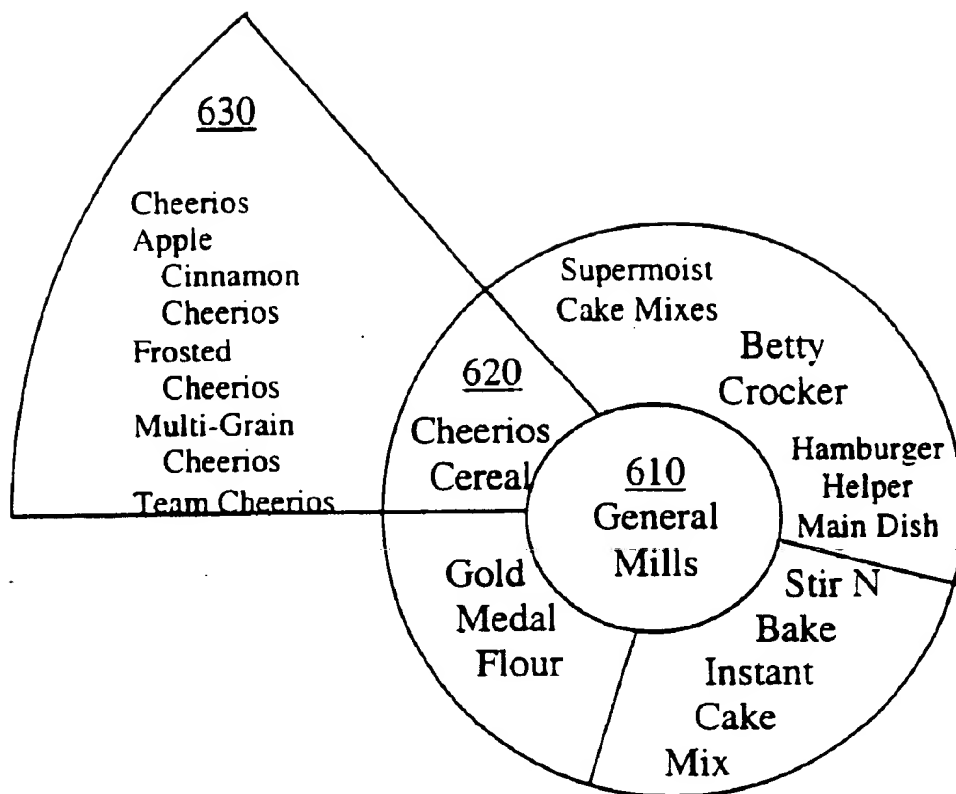


Fig. 6C

**Fig. 6D**

**Sample Rules for The Brand Architecture Chart of Figure 6A****Product Family Brands**

- Multiple products at launch
- Minimum \$50 million launch budget in year 1
- Approval: Executive VP of Marketing and CEO

**Products**

- Appropriate for flavors, form factors, or other extensions of Product Family brands
- Descriptive terminology only
- Approval: automatic if descriptive terminology employed
- May only consider proprietary (trademarked) sub-brand name (e.g., Softasilk) if:
  - Minimum \$20 million launch budget in year 1
  - Potential to line extend sub-brand, such that it could eventually grow into a Product Family brand
  - Approval: Executive VP of Marketing

**Special Ingredients**

- Must employ the following format: "with [ingredient]"
- Use of ingredient name must be expected to add at least 30% to sales of relevant Product Family brand
- Ingredient must not be greater in prominence than General Mills or Product Family brand, as measured by annual advertising budgets
- Approval: Executive VP of Marketing

**Figure 7**

1

# COMPUTER IMPLEMENTED METHOD OF IMPLEMENTING, MAINTAINING, AND UPDATING A BRAND ARCHITECTURE SYSTEM

## FIELD OF THE INVENTION

The present invention relates to brand architectures, and more specifically, to an automated brand architecture administration system.

## BACKGROUND

A brand name is a name or symbol used to identify a seller's goods or services, and to differentiate them from those of competitors. Because a brand name identifies a product's or service's source—protecting against competitors who may attempt to market similar goods or services—companies have an incentive to invest in the quality, consistency, and imagery of their brands.

In the prior art, often there was no unified method to manage brand names within a company. In one prior art system, trademark counsel kept a list of trademarks and trademark filings for the purposes of keeping trademarks up-to-date and renewing them. Product managers in the organization created new products and potential names for them. The product managers then requested a search for trademark availability of the names. Other companies did not do any searching, causing the companies to risk being sued for trademark infringement.

Generally, in the prior art, organizations had no marketing or strategic central determiner for whether a product needed a name or whether the name chosen by the product manager was appropriate. Furthermore, each time an additional product or service was created the trademark counsel and product manager had to individually process the proposed name.

Therefore, it would be advantageous to reduce the complexity of administering a company's brand names.

## SUMMARY OF THE INVENTION

A method of automatically managing and maintaining a brand architecture is described. The method comprises displaying a navigable brand-architecture chart. The method further comprises helping users identify the appropriate, approved location of the desired name on the brand architecture chart. The method further includes testing the name against rules, to see whether the name fits the criteria. If the new name fits criteria, for one embodiment the proposed name is automatically sent to an approver, or to a legal department for approval. For another embodiment, the brand architecture chart is automatically updated with the new name.

For one embodiment, if the name does not fit the criteria, an appeal process to higher tier managers may be automatically triggered. For one embodiment a request for additional conforming name candidates is generated and sent to the appropriate party or parties.

## DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

FIG. 1 is a block diagram of one embodiment of a network on which the present invention may be used.

2

FIG. 2 is one embodiment of computer system on which the present invention may be implemented.

FIG. 3 is a flowchart of one embodiment of the method of editing the brand architecture chart.

FIG. 4 is a block diagram of one embodiment of the branding software.

FIG. 5 is a sample filled-out questionnaire.

FIG. 6A is an example of a possible layout for a brand architecture chart.

FIG. 6B is an example of a possible layout for displaying a portion of the brand architecture chart of FIG. 6A.

FIG. 6C is another example of a possible layout for a brand architecture chart.

FIG. 6D is an example of a possible layout for displaying a portion of the brand architecture chart of FIG. 6C.

FIG. 7 is an example of the rules implemented by the sample brand architecture charts of FIGS. 6A–D.

## DETAILED DESCRIPTION

In the present invention, a brand architecture system is described. A brand architecture specifies and organizes the naming relationships among a company's brands, products, services, divisions, subsidiaries, etc. A well-conceived brand architecture will accommodate company and product line growth, and provide guidance for future product and service names. Some brand architectures are comprised of many tiers, with specific naming guidelines for each.

The brand architecture system described includes a navigable brand-architecture chart. A user is permitted to nominate a new name including a desired location on the brand-architecture chart to the system. For one embodiment, the user may be anyone with authority to access the brand architecture chart. For one embodiment, employees at different tiers have different tiers of access, from viewing access only, to name proposal access, to alteration access.

The proposed new name is tested against the naming rules at that location, to see whether the name fits criteria. If the new name fits the criteria, for one embodiment upon the approval of the legal department and/or a designated approving authority, the brand architecture chart is automatically updated with the new name. Alternatively, the brand architecture chart may be updated without the specific approval, based on the fit of the proposed name into the chart. For an alternative embodiment, the new name is added to a list of approved names, and the brand architecture chart is updated by an individual with alteration access.

For one embodiment, if the name does not fit the criteria, alternative allowable naming parameters are suggested. For one embodiment, if the name not fit the criteria, an appeal process to higher tier managers is automatically triggered. For one embodiment, request for additional name candidates is also sent to the appropriate parties.

FIG. 1 is a block diagram of one embodiment of a network on which the present invention may be used. An internal network server 110 may be set up on a client's system. The internal network server 110 is accessed by a plurality of clients.

The internal network server 110 accesses a system 130 on which the brand architecture software is implemented through a network 120. For one embodiment, this system 130 may be referred to as the brand architecture system 130, although the server may not be a dedicated server, and the brand architecture software may be on an existing LAN/Intranet. For one embodiment, the internal network server

3

110 may be the same computer as the brand architecture server 130. For another embodiment, the internal network server 110 may be a separate computer, and the network 120 may be an internal large area network (LAN), wide area network (WAN), the Internet or other network. For yet another embodiment, the brand architecture server 130 may be outside the company, and in fact may be administered by someone outside the company. In that case, the network 120 may be the Internet. For one embodiment, if the network 120 is the Internet, communications between the internal network server 110 and the brand architecture server 130 may be secured by using secure hypertext transmission protocol (https), by encryption, or through some other means of securing the communication. The brand architecture server 130 includes brand architecture software 140, which is described in more detail below.

FIG. 2 is one embodiment of computer system on which the present invention may be implemented. FIG. 2 illustrates a typical data processing system upon which one embodiment of the present invention is implemented. It will be apparent to those of ordinary skill in the art, however that other alternative systems of various system architectures may also be used.

The data processing system illustrated in FIG. 2 includes a bus or other internal communication means 245 for communicating information, and a processor 240 coupled to the bus 245 for processing information. The system further comprises a random access memory (RAM) or other volatile storage device 250 (referred to as memory), coupled to bus 245 for storing information and instructions to be executed by processor 240. Main memory 250 also may be used for storing temporary variables or other intermediate information during execution of instructions by processor 240. The system also comprises a read only memory (ROM) and/or static storage device 220 coupled to bus 240 for storing static information and instructions for processor 240, and a data storage device 225 such as a magnetic disk or optical disk and its corresponding disk drive. Data storage device 225 is coupled to bus 245 for storing information and instructions.

The system may further be coupled to a display device 270, such as a cathode ray tube (CRT) or a liquid crystal display (LCD) coupled to bus 245 through bus 265 for displaying information to a computer user. An alphanumeric input device 275, including alphanumeric and other keys, may also be coupled to bus 245 through bus 265 for communicating information and command selections to processor 240. An additional user input device is cursor control device 280, such as a mouse, a trackball, stylus, or cursor direction keys coupled to bus 245 through bus 265 for communicating direction information and command selections to processor 240, and for controlling cursor movement on display device 270.

Another device that may optionally be coupled to computer system 230 is a communication device 290 for accessing other nodes of a distributed system via a network. The communication device 290 may include any of a number of commercially available networking peripheral devices such as those used for coupling to an Ethernet, token ring, Internet, or wide area network. Note that any or all of the components of this system illustrated in FIG. 2 and associated hardware may be used in various embodiments of the present invention.

It will be appreciated by those of ordinary skill in the art that any configuration of the system may be used for various purposes according to the particular implementation. The control logic or software implementing the present invention

4

can be stored in main memory 250, mass storage device 225, or other storage medium locally or remotely accessible to processor 240. Other storage media may include floppy disks, memory cards, flash memory, or CD-ROM drives.

It will be apparent to those of ordinary skill in the art that the methods and processes described herein can be implemented as software stored in main memory 250 or read only memory 220 and executed by processor 240. This control logic or software may also be resident on an article of manufacture comprising a computer readable medium having computer readable program code embodied therein and being readable by the mass storage device 225 and for causing the processor 240 to operate in accordance with the methods and teachings herein.

The software of the present invention may also be embodied in a handheld or portable device containing a subset of the computer hardware components described above. For example, the handheld device may be configured to contain only the bus 245, the processor 240, and memory 250 and/or 225. The handheld device may also be configured to include a set of buttons or input signaling components with which a user may select from a set of available options. The handheld device may also be configured to include an output apparatus such as a liquid crystal display (LCD) or display element matrix for displaying information to a user of the handheld device. Conventional methods may be used to implement such a handheld device. The implementation of the present invention for such a device would be apparent to one of ordinary skill in the art given the disclosure of the present invention as provided herein.

FIG. 3 is a flowchart of one embodiment of the method of accessing and querying the brand architecture chart. At block 300, the process starts. The process starts when an authorized user accesses the software, and indicates that he or she wishes to access the brand architecture, to learn about it, to test a name candidate against it, or to add a new item to the architecture.

At block 305, a tutorial is displayed. For one embodiment, the tutorial may be optionally skipped. For one embodiment, this step may be omitted entirely. The tutorial may teach a branding philosophy and/or strategy of the company, the objectives of the branding architecture. The tutorial may include the company's brand architecture policies, methodologies, and rules. For one embodiment, the tutorial may further teach how the branding structure of the company works. For one embodiment, the tutorial may be interactive. Alternatively, the tutorial may be text based, video based, audio based, or in an other format.

At block 310, a navigable brand architecture chart is displayed to the user. Sample charts are shown in FIGS. 6A-D. For one embodiment, this chart may be large enough to cover multiple pages or screens.

For one embodiment, the chart is displayed hierarchically, i.e. only the product family names are displayed, and the user can select to display the lower tier information for each family. For another embodiment, another method of displaying the chart may be used. FIGS. 6B and 6D illustrate another embodiment of the chart, permitting access to all tiers of the chart, while displaying only the relevant portions of the chart.

For one embodiment, the user may access the brand architecture chart without initiating the editing process. In that instance, the process stops here, and the user can view the navigable brand architecture chart. If the user initiates the editing process, the process continues to block 315.

At block 315, the user is prompted to choose a specific tier of the chart they wish to view. For one embodiment, this is

5

a selection from a pull-down menu or by clicking the tier in the chart. For another embodiment, the user types in the tier name. For yet another embodiment, the user is not directly prompted for this information, but is rather asked to fill in a questionnaire from which this information is deduced. A sample of the questionnaire is included in FIG. 5. For another embodiment, another means of selecting the tier may be used. For one embodiment, this tier may be a "product family" as shown in FIGS. 6A-D. Alternative organizations of brand architecture charts may have alternative tiers, and selections.

At block 320, the navigable chart of the selected tier is displayed to the user. This is a subset of the brand architecture chart displayed at block 310. For embodiment, this is the same chart, with the colors altered, to indicate the selected tier.

At block 325, the user is prompted to choose a product tier. As above, this choice may be made in a variety of ways, including automatically based on the questionnaire.

At block 330, the tier and product tier chosen is displayed to the user.

At block 335, the user is prompted to enter a proposed name.

At block 340, it is determined whether the proposed name fits into the rules of the selected family and tier. As will be discussed in more detail below, the rules restrict the types of names that will be acceptable at each tier. The purpose of the rules is to ensure the implementation of the company's adopted brand architecture, and simplify naming options. These rules are discussed in more detail below, with respect to FIG. 7. If the proposed name fits into the rules, the process continues to block 370. At block 370, the user is informed that the proposed name will be added to the chart.

For one embodiment, the chart is automatically updated at this stage. If, for example, only those with naming authority within the company have access to this process, automating the updating of the chart may be advantageous. The process then ends at block 375. For another embodiment, the approved name is added to a list of proposed names. An authorized person with authority to update the chart then can update the actual chart. For another embodiment, the proposed name may be sent to the legal department or other appropriate person for approval. For example, the company may wish to review a name, even if it fits the rules, to verify that the name is appropriate, that there are no trademark issues, and/or that there are no other criteria that are not embodied in the brand architecture chart.

If the proposed name did not fit into the rules, at block 340, the process continues to block 345. At block 345, it is determined whether the user wishes to proceed with the name in spite of this. For one embodiment, this is determined by prompting the user. For one embodiment, the user should meet certain criteria in order to proceed at this stage. For example, the user may be asked to commit a certain amount of money from his or her budget to search and/or provide marketing support for the proposed name. Other indications of the user's commitment to the name may also be requested. If the user does not wish to go ahead with the name, the process returns to block 335, and the user is again prompted to enter a proposed name. If the user wishes to go ahead with the name, the process continues to block 350.

At block 350, the user is prompted to enter reasons why the non-conforming name should be approved. For one embodiment, this may be done in the form of a questionnaire, or a free-form data entry. For one embodiment, the user provides projections for the product

6

that is being named, including potential sales, etc. For one embodiment, the user's answers to the questions will determine whether the name is likely to be approved although it is outside the approved architecture.

At block 355, the user's entry is forwarded to a designated individual or group who have authority to override the rules embodied in the brand architecture. For one embodiment, this step automatically generates an e-mail sent to the authorized person or persons, including the user's name suggestion, why it does not conform to the branding architecture, whether override criteria have been met, and any other relevant information. Designated persons are authorized to approve the override. For one embodiment, this step automatically connects to a organization chart of the company, and determines correct person or persons to receive this e-mail. For one embodiment, the correct person may be a branding executive, a marketing executive and/or the intellectual property legal department.

For one embodiment, the user is disconnected from this process at this stage. For one embodiment, the user is informed that the request is being forwarded to the person or persons authorized to approve the override request. For one embodiment, the time for approval may be days or weeks.

At block 360, it is determined whether the name has been approved. For one embodiment, the authorized person may return an affirmative or negative statement directly to the brand architecture software. The brand architecture software then analyzes the response, and determines whether it was positive or negative. For one embodiment, if multiple people have authorization to override, either a subgroup of or all of the group must return an affirmative answer in order to end with an approval.

For another embodiment, if the system does not receive a response, either affirmative or negative, within a preset period, the system sends a reminder request to the authorized persons.

If an affirmative response is received, the process continues to block 370, and the user is informed that the requested name has been approved and the proposed name will be added to the chart, and the chart is updated.

If a negative response is received, the process continues to block 365, and the user is informed of the unfavorable response. For one embodiment, the user is also provided a pointer to follow, to reapply for a new name. For one embodiment, the user may follow this pointer, and not have to select the product family and product tier, but rather go directly to block 335, to propose a new name. For one embodiment, the user can request help developing alternative names. A request is then sent to the appropriate name development person or outside agency.

The process terminates at block 375.

FIG. 4 is a block diagram of one embodiment of the brand architecture software. The brand architecture system 410 includes a brand architecture chart display unit 420. The brand architecture chart display unit 420, for one embodiment, is capable of displaying a multi-page brand architecture chart, in various formats. For one embodiment, the format may be a hypertext markup language (HTML) format, a JavaScript implemented format, or another display format. For one embodiment, the chart display unit 420 permits navigation along a family, tier, or arbitrary axis.

The chart configuration unit 430 controls which parts of the chart are displayed by the display unit 420. For one embodiment, as a user navigates through the chart, a narrower and narrower slice of the chart may be visible. The chart configuration unit 430 controls this, based on data from



the user interface. If, for example, as described above, the user selects a particular product family, the chart configuration unit 430 instructs the chart display unit 420 to display only the selected product family.

The brand architecture system 410 further includes a user interface 440. The user interface 440 allows a variety of users to access the chart. An access control unit 450 is coupled to the user interface. The access control unit 450 determines the access tier of each user. For one embodiment, the access control unit 450 includes a plurality of classifications for potential users of the system 410. For example, there may be trusted users, those with authority to actually update the system, authorized users, those with authority to propose new names for inclusion in the chart, and other users, those who may look at the system, but may not change it. For another embodiment, the access control unit 450 may use some other method, such as a list of names, to determine access tiers.

For one embodiment, the user interface 440 is configured to the appropriate tier for a user, once that user accesses the system 410. For one embodiment, the user interface 440 displays only those options that are available to the user.

A comparison unit 460 is coupled to the user interface 440. When the user interface 440 receives a proposed name, for a selected product family and product tier, this information is passed to the comparison unit 460. The comparison unit 460 accesses the rules for this particular product family and product tier in the rule storage unit 460. If the name passes the rules, the user is informed of this. For one embodiment, the rules are displayed to the user by the comparison unit 440. If the proposed name meets rules, the name is submitted for approval and entry into the brand architecture chart. If the proposed name is a non-conforming name, the proposed name is forwarded to the override facility 480, if the user so requests.

For one embodiment, the comparison unit 460 automatically determines whether the name as proposed passes certain rules. Certain of the rules may be automatically tested by comparison unit 460. For example, if the rule states that all product names must start with the letter "N," the comparison unit 460 may test whether the proposed name passes this rule. Similarly, for rules such as incorporating certain words, or family brand names, into the proposed name, having a certain number of letters in a name, etc. the comparison unit 460 may automatically determine whether the rule is met by the proposed name.

For one embodiment, for certain rules, the user is merely made aware of the rule and the user has to determine whether the rule is met by the proposed name. For example, for a rule such as "the name must be descriptive" the comparison unit 460 may not test the proposed name, for one embodiment. Rather, the user determines whether the proposed name meets the rule, or whether the override facility should be invoked.

For one embodiment, if the comparison unit 460 determines that the name does not meet the rules, the user is informed of this. For another embodiment, if the comparison unit 460 determines that the name does not meet rules, this is highlighted for the approving authority.

If the name does not pass the rules, the user may ask for assistance for generating alternate names. The a request may automatically be sent to a naming company, or other agency internal to or external to the company, to assist the user in generating an appropriate name. For one embodiment, the comparison unit 460 permits the user to generate this message, and passes the message to the appropriate organizations and/or individuals.

If the name did not pass the rules, but the user confirms that he or she wishes to go ahead with the name, the comparison unit 460 passes the information to the override facility 480.

The override facility 480 automatically forwards the override request to an authorized approver. For one embodiment, the override request is e-mailed to the authorized approver. If the authorized approver is not within the secure network on which the brand architecture system 410 resides, the e-mail or other notification may be encrypted or otherwise secured.

The override facility 480 receives the response(s) from the authorized approver(s). For one embodiment, the override facility 480 sends a reminder or other notification if no response is received within a set period. When the response, or all of the responses, are received, the override facility determines whether the response(s) are positive or negative. For one embodiment, if there are multiple responses, the system can be set to require positive responses from all approvers, or from only a certain number of approvers, or from only a certain approver. The override facility 480 returns the affirmative or negative indication to the comparison unit 460.

The comparison unit 460 forwards this information to the user, and to the chart updating facility 490. If the name was approved, the chart updating facility 490 updates the chart displayed by the display unit 420. For one embodiment, the chart updating facility 490 can automatically update the chart if the nominated name either meets the rules, meets the rules and has been approved, or if the override request is approved. For another embodiment, the chart updating facility 490 maintains approval information. Then, the next time a user with authority to alter the chart logs in, this information is made accessible to the user, and the user can add the approved names.

FIG. 5 is a sample questionnaire. The sample questionnaire is drawn up to correspond to the structure illustrated in FIGS. 6A-D. Of course, as the brand architecture changes for each company, the questionnaire would change correspondingly. It is to be understood that FIG. 5 is exemplary, and should not be interpreted to be a complete or necessary format for such a questionnaire.

A similar type of questionnaire may be displayed to a user attempting to nominate a new name for a new product, service, feature, or technology. The flowchart described in FIG. 3 may be automatically processed—the selection of the appropriate category for the name, the name selection, as well as the method of going outside the rules—based on the information filled into the questionnaire.

The general information 510 that is requested from the user may include the user's name, title, group, and contact information. For one embodiment, the general information 510 may be automatically filled in based on the user's log-in into the system. However, the user is permitted, in one embodiment, to alter the data in the general information category 510.

The first tier selection 520 permits the user to select a brand, product, or category for the new name, in the example chart of FIGS. 6A-D. The user can select whether the new item will be a product family brand, a product, or an ingredient or feature.

The second tier selection 530 permits the user to select a subcategory for the new name. For example, in the brand architecture of FIGS. 6A-D, if the user selected a Product as the first tier selection for the proposed name, the user at this point can select whether the proposed name should be added

into the Cheerios Cereal product family, the Super Moist Cake Mix family, etc. After this selection is complete, the system knows what the category and subcategory location is for the new name. Of course, this type of categorization may continue. For example, included in the Cheerios Cereal product family, there may be further sub-categories, such as Children's Cheerios and Adult Cheerios or Low Salt Cheerios and Standard Cheerios. Any such sub-categorizations could be consecutively selected by the user. The user may select either to the lowest tier, i.e. the lowest subcategory available, or the user may suggest the name for a higher tier. For example, in the brand architecture of FIGS. 6A-D, the user may propose a new product family, such as Baking Ingredients.

The product/service definition category selection 540 permits the user to enter a description. This may be optional, unless the user wishes to select a name outside the rules for the selected location.

The proposed name 550 is the new name the user wishes to add to the brand architecture. For one embodiment, the user may review the rules for the proposed location for the name. For one embodiment, the user may be able to view the list of current names within the selected location, in order to simplify the name selection. For another embodiment, the user may see examples of approved names.

For one embodiment, a secondary questionnaire page may be shown to the user if the user's first proposed name has been found to be non-conforming name. For one embodiment, these questions are always shown to the user, but the user is only specifically prompted to fill in the answers if the name is a non-conforming name.

The user's commitment to the product 560 may include a certain amount of money from a budget to research and/or provide marketing support for the name. Other indications of commitment, such as manpower, advertising budget may also be requested. For one embodiment, the commitment tier 560 is automatically requested for product family brands. For one embodiment, this question is only displayed for lower categories if the name was found to be a non-conforming name and the user wishes to submit the name for extra-ordinary authorization.

The projected income from the product 570 is similar to the commitment tier. For one embodiment, the projected income is automatically requested for certain tiers, such as product family brands and special ingredients, but is not requested for lower tiers, unless the proposed name is a non-conforming name.

The request for help naming the product 580 may also be displayed. For one embodiment, the request for help is displayed only when the user's initial name is rejected. For another embodiment, the request for help may also be displayed if the proposed name is a non-conforming name and is likely to be rejected, or if a user is attempting to overcome a rejection. For one embodiment, the request for help 580 sends an e-mail or other communication to an outside agency or to a naming group within the corporation, to assist the user in selecting an appropriate name. For one embodiment, this is a discontinuous process, such that the user is disconnected from the brand architecture system, and is separately contacted with naming help.

FIG. 5 illustrates one example a questionnaire that may be used to guide a user through the naming process. For an alternative embodiment, the user may make all of the above selections through navigating web pages or computer display screens and the brand architecture chart, and using menus. These types of selections among choices and entry of data may be used, and are well known in the art.

FIG. 6A is an example of a brand architecture chart that may be displayed by the present invention. The brand architecture chart of FIG. 6A is only a sample chart, implementing sample rules. It is to be understood that the methods used to generate this chart may be used to generate more or less complex charts, and implement dissimilar rules. The format of the chart may also vary from the format shown in FIG. 6A. FIG. 6A is merely one possible display of one sample chart. FIG. 6C illustrates an alternative display for the brand architecture chart. It is to be understood FIGS. 6A-D illustrate exemplary displays of the brand architecture chart.

This brand architecture chart is an example of a chart that may be viewed, queried, manipulated, and updated by the invention, as described above. The General Mills product line is shown as an example. No affiliation of this patent is meant or implied by using the General Mills product line as an example. All trademarks mentioned in the chart and in the text below are the property of their respective owners.

The first tier 610 is the corporate name 610, which is General Mills.

The second tier 620, are product family names. In this example, product family names 620 include a number of different brands, for example Cheerios cereals, Betty Crocker's Supermoist Cake Mix, Hamburger Helper, Stir N Bake, and Cold Medal Flour. In this example, the product family names 620 may be fanciful names, such as Cheerios, or descriptive names, such as Super Moist Cake Mix. In other examples, a wide variety of other types of names may be allowed.

For a different company, the second tier 620 may not be product family names, but rather, for example, service types, or targeted customers, or another type of category. The selection of the actual categories for the brand architecture chart is not made by the present system, but rather implemented by a person or persons who are familiar with the brand architecture of the actual corporation.

The third tier 630, in this case, is specific products. Thus, in each product family brand 620 there are a number of specific products 630. Thus, for example, the Stir N Bake product family brand 620 includes Brownies, Carrot Cake, and other products. In this example, the products 630 generally have descriptive names. For some product family brands 620, e.g. Cheerios, the product names 630 each include the product family brand name.

Again, in a different corporation, this tier may have a different name, and different rules, and different types of names or name relationships may be allowed.

The fourth tier 640, in this case, is special ingredients or features. Certain products 630 may further include special ingredients 640. For one embodiment, the special ingredients 640 may be items are named on the package, or otherwise highlighted. As can be seen, the special ingredients 640 in this instance are descriptive names of the actual additional ingredient.

As discussed above, these tiers and the items within each tier are entirely dependent on the actual structure of the corporation. For example, in a service corporation, the categories would be different, as would the types of items within each category. Similarly, the rules implemented at each tier may differ substantially depending on the actual corporation. The above list of products and product organizations has been derived independently based on products currently available for a company. It is not to be taken as an actual representation of the products or brand architecture of the company in question. Furthermore, this does not imply any connection between the patentee and the company.

FIG. 6B illustrates a portion of the brand architecture chart of FIG. 6A. FIG. 6B shows a sample display format for displaying a large brand architecture chart that would generally not fit on a single page. The user can select one of the arrows 650, for example, selecting the Super Moist Cake Mix product family brand 620. In that instance, the products listed within this product family brand 620 would be displayed. For one embodiment, the higher categories are not shown, and only the current category, i.e. product family brand, product, or special ingredients is displayed. For another embodiment, once a product family brand 620 is selected, the lower and higher tiers associated with the selected product family brand 620 are displayed. Thus, for example, if the user selects the Cheerios Cereal product family brand 620, the display would include the Corporate name, the Cheerios Cereal product family brand 620, and the products 630 (not shown) that belong in the Cheerios Cereal Product Family brand 620. In this way, the user can easily navigate the brand architecture chart, while only seeing the relevant category at any time.

FIG. 6C illustrates another alternative method of displaying a brand architecture chart. The brand architecture chart is shown as a circular structure, with the corporate name at the center, product family brands in the first ring, products in the second ring, and special ingredients in the third ring. It is to be understood that other brand architecture charts, not having similar components, could similarly be shown in a circular structure.

FIG. 6D illustrates a portion of the brand architecture chart of FIG. 6C. FIG. 6D shows a sample display format for displaying a large brand architecture chart that would generally not fit on a single page. The user can select one of the arrows 650, for example selecting the Cheerios Cereal product family brand 620. In that instance, the products 630 listed within this product family brand 620 would be displayed. For one embodiment, the higher categories are not shown, and only the current category, i.e. product family brand, product, or special ingredients is displayed. For another embodiment, once a product family brand 620 is selected, the lower and higher tiers associated with the selected product family brand 620 are displayed. Thus, for example, if the user selects the Cheerios Cereal product family brand 620, the display would include the Corporate name, the Cheerios Cereal product family brand 620, and the products 630 that belong in the Cheerios Cereal Product Family brand 620. In this way, the user can easily navigate the brand architecture chart, while only seeing the relevant category at any time.

It is to be understood that FIGS. 6A and 6C are exemplary displays of brand architecture charts. Alternative display formats, such as three dimensional triangular, diamond shaped, or having some other shape suited to the particular brand architecture may be used.

FIG. 7 shows a sample set of rules that are implemented in the brand architecture chart of FIGS. 6A-D. The rules may include naming conventions, such as the form of the proposed name, the syllables in the name, the length of the name, the distinctiveness of the name, and any other rules the corporation may wish to implement. The rules may further include information and/or requirements regarding the use of logos, colors, typefaces, and graphics.

The rules may include such rules as the number of products in a product family at launch, the budget, etc. For one embodiment, certain categories, such as product family and proprietary product names may automatically require the approval of someone. In these instances, the flowchart automatically follows the "name rejected" route, described above.

It is to be understood that these rules are only exemplary rules, and different rules may be implemented. Additional rules may be implemented as well, including such rules as requiring a trademark search for a new name, requiring certain persons to support a new name, etc. Other rules may be specific to the corporation—for example, a rule may be that new product names must start with the letter T. Any rules that can be imagined may be implemented in the rules for the brand architecture chart. In this manner, the approval of new names and the adding of new names to the brand architecture chart to form a coherent branding strategy may be automated.

In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. Each brand architecture solution will vary more or less from these examples. Similarly, the display formats and the rules that are implemented vary based on the actual structure of the corporation and the brand strategy of the corporation. The focus of the present invention is the automatic requests, updating, and referrals, as described above. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

What is claimed is:

1. A computer-implemented brand architecture system comprising:
  - a hierarchical brand-architecture chart comprising:
    - at least one group displayed at a first tier;
    - a plurality of items displayed at a second tier, each of the plurality of items at the second tier categorized under at least one group of the first tier; and
    - an updating facility for updating the navigable brand-architecture chart, such that a new entry is entered in a proper group of the plurality of groups, and at a proper tier.
2. The system of claim 1, wherein the plurality of tiers comprise:
  - a first tier including a plurality of product families;
  - a second tier including a plurality of individual products, wherein each of the individual products are categorized in one of the product families.
3. The system of claim 2, wherein the second tier includes a product group having multiple individual products within the product group.
4. The system of claim 1, further comprising:
  - a chart configuration unit for controlling the amount of the chart that is displayed.
5. The system of claim 1, further comprising:
  - a comparison unit, for receiving a proposed name, and for comparing the proposed name against a set of rules, and determining if the proposed name fits within the rules.
6. The system of claim 5, further comprising:
  - an override facility, the override facility for requesting permission to override the rules, and permit the proposed name that failed the rules.
7. The system of claim 6, wherein requesting permission comprises automatically sending a request to authorized personnel, and determining whether the response was affirmative or negative.
8. The system of claim 1, further comprising:
  - an access control unit for limiting access to the brand architecture chart, the access control unit permitting only certain users to alter the chart.

## 13

9. The system of claim 8, wherein the access control unit permits certain users to alter the chart, and certain other users to request a new name.

10. A method of implementing an automated brand architecture management system, comprising:

displaying a hierarchical brand-architecture;

receiving a nomination for a new name including information indicative of a position in the hierarchy of the brand-architecture;

comparing the nomination against rules to determine whether the new name fits criteria; and

if the new name fits criteria submitting the nomination for approval and, automatically updating the brand-architecture with the new name upon receipt of approval.

11. The method of claim 10, wherein if the new name does not fit the criteria,

determining if there is an exception procedure; and

if there is an exception procedure, implementing the exception procedure.

12. The method of claim 11, wherein the exception procedure comprises:

automatically electronically sending the nomination and accompanying information to an authority who can approve the exception to the rules.

13. The method of claim 12, further comprising:

receiving a response from the authority; and

if the response is affirmative, automatically updating the brand architecture chart, and informing a nominator of the affirmative response.

14. The method of claim 13, further comprising:

if the response is negative, informing a nominator of the negative response, and permitting the nominator to nominate a new proposed name for the product.

15. The method of claim 13, further comprising:

if the response is negative, informing a nominator of the negative response and permitting the nominator to request help generating a proposed name that would result in a positive response.

16. The method of claim 13, wherein the authority is a plurality of persons, and wherein the response comprises a plurality of responses, and an affirmative response comprises affirmative responses from each of the plurality of persons.

17. The method of claim 11, wherein the exception procedure comprises:

requiring a commitment from the nominating group to support the new name.

18. The method of claim 10, wherein the rules comprise a limitation on a type of name being used.

19. The method of claim 10, wherein the step of receiving a nomination for a new name including a location in the brand-architecture chart, comprises:

receiving a filled-in questionnaire, indicating a tier and a category, and a proposed name.

20. The method of claim 19, wherein after the user selects a product family and a product tier, the rules for naming within that product family and product tier are displayed to the user, prior to allowing the user to enter the nomination.

21. The method of claim 20, further comprising indicating to the user that the name may be outside of the rules, if certain conditions are met.

22. The method of claim 10, wherein the step of receiving a nomination for a new name comprises:

prompting a user to select a product family, after reviewing the entire brand chart;

## 14

prompting the user to select a location, after reviewing the entire product family; and

prompting the user to enter the nomination for a proposed product name.

23. A method of maintaining a brand architecture management system to simplify maintaining coherent branding strategy, the method comprising:

defining a brand architecture chart including current and proposed names;

defining a rule set for the brand architecture chart to maintain the coherent branding strategy;

incorporating the rule set into an automatic nomination system that receives new name nominations from users; and

prompting a user to compare a new name nomination against the rule set, and accepting the new name nomination if the new name fits within the rule set.

24. The method of claim 23, wherein if the new name nomination does not fit within the rule set, the method further comprises:

permitting the user to request help in selecting a different name; and

permitting the user to appeal the decision to an authority that may override the rule set.

25. The method of claim 23, further comprising automatically adding the new name to the brand architecture chart if the new name is accepted.

26. An automated brand architecture management system comprising:

a rule storage unit for storing pre-determined rules;

a display mechanism for displaying a hierarchical brand-architecture;

a user interface for receiving a nomination for a new name from a user, the nomination including information indicative of a position in the hierarchy of the brand-architecture;

a comparison unit for comparing the nomination against the pre-determined rules from the rule storage unit, to determine whether the new name fits criteria and submitting the nomination for approval if the new name meets the pre-determined rules; and

a chart updating facility for automatically updating the brand-architecture with the new name upon receipt of approval.

27. The automated brand architecture management system of claim 26, further comprising, if the new name does not fit the criteria:

an override facility for implementing an exception procedure.

28. The automated brand architecture management system of claim 27, wherein the exception procedure comprises the override facility automatically electronically sends the nomination and accompanying information to an authority who can approve the exception to the rules, if the new name.

29. The automated brand architecture management system of claim 28, further comprising:

the override facility for receiving a response from the authority; and

if the response is affirmative,

a chart updating facility for automatically updating the brand architecture chart, and informing a nominator of the affirmative response.

30. The automated brand architecture management system of claim 29, further comprising:

15

if the response is negative,  
the user interface for informing a nominator of the negative response, and permitting the nominator to nominate a new proposed name for the product.

31. The automated brand architecture management system of claim 29, further comprising:

if the response is negative,  
the user interface for informing a nominator of the negative response; and  
an access control unit for permitting the nominator to request help generating a proposed name that would result in a positive response.

32. The automated brand architecture management system of claim 28, wherein the authority is a plurality of persons, and wherein the response comprises a plurality of

16

responses, and an affirmative response comprises affirmative responses from each of the plurality of persons.

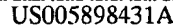
33. The automated brand architecture management system of claim 26, wherein the exception procedure comprises:

the override facility requiring a commitment from the nominating group to support the new name.

34. The automated brand architecture management system of claim 26, wherein the rules comprise a limitation on a type of name being used.

35. The automated brand architecture management system of claim 34, wherein after the user selects a tier, the user interface displays rules for naming within that tier, prior to allowing the user to enter the nomination.

\* \* \* \* \*



## [19]

[11]

[45]

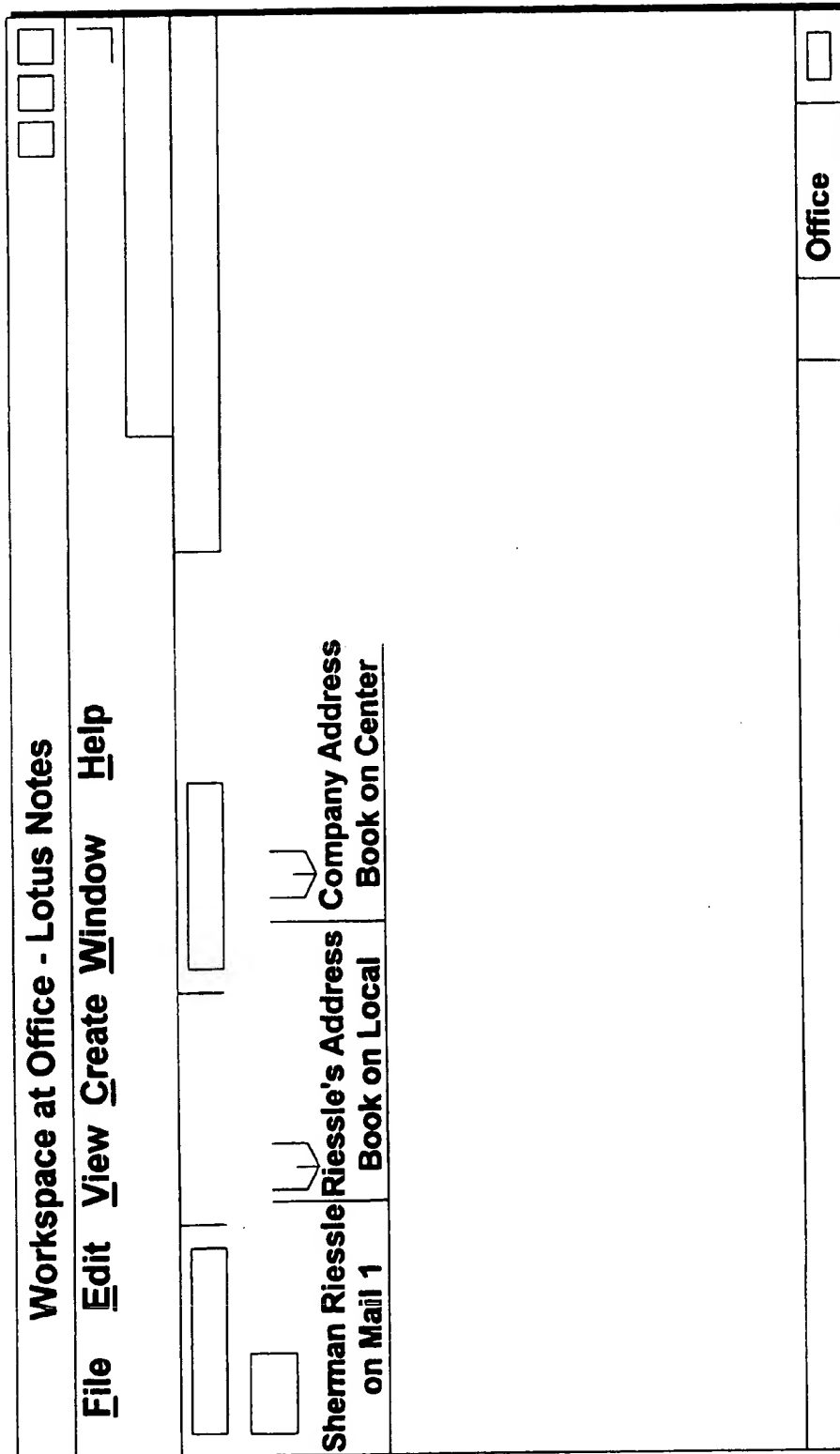
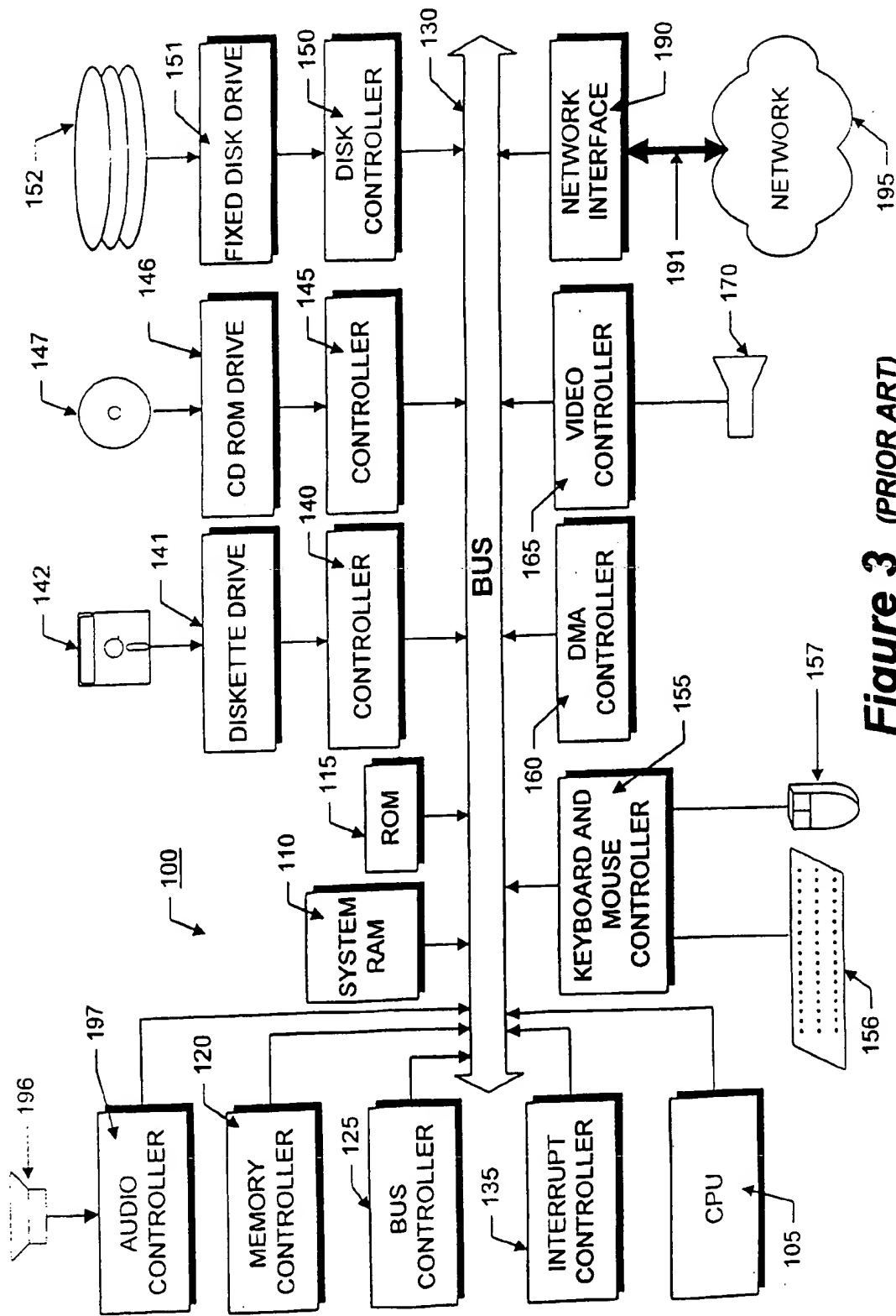


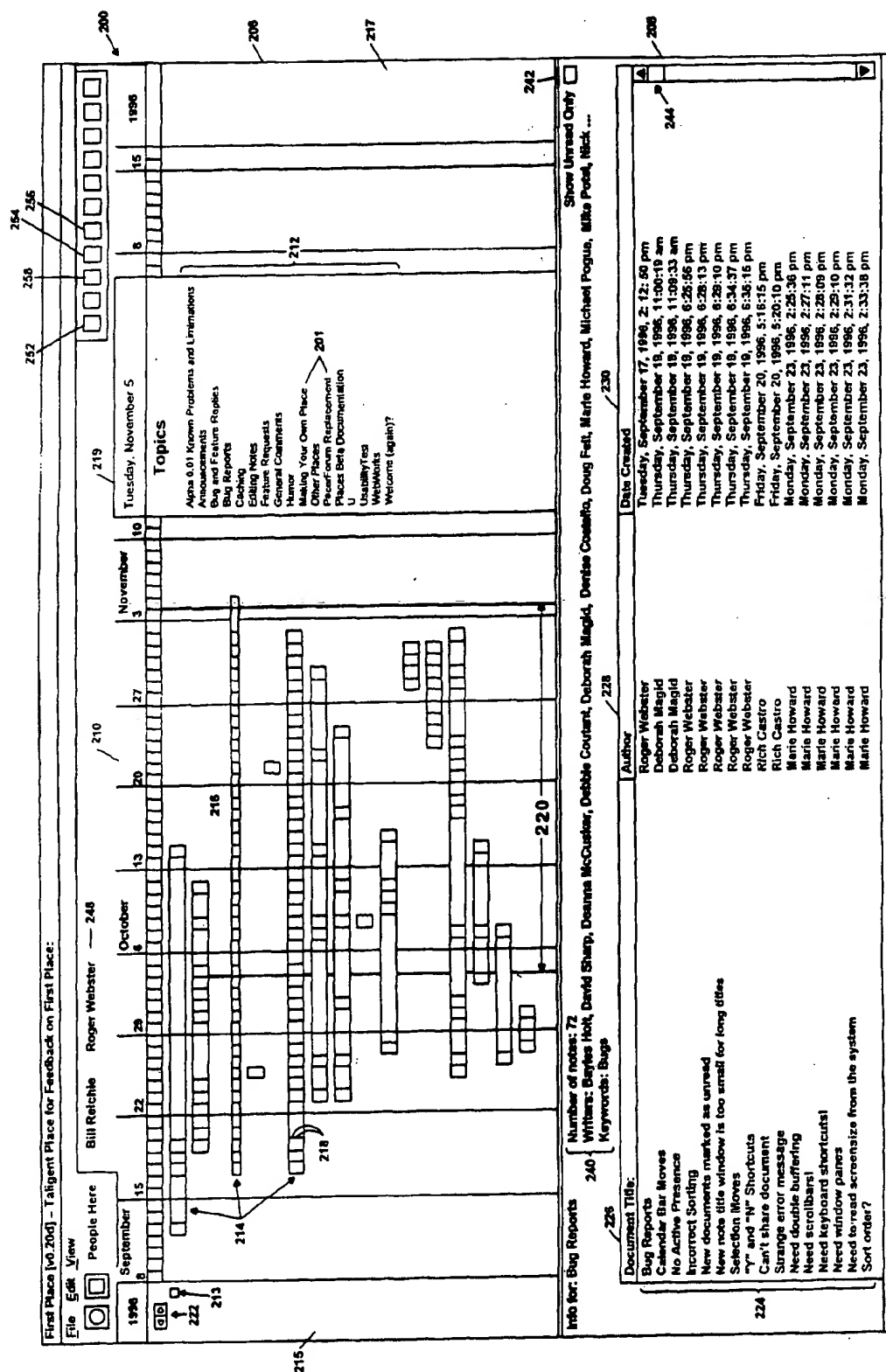
Figure 1

Company News - All Documents - Lotus Notes		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>C</u> reate <u>A</u> ctions <u>W</u> indow <u>H</u> elp	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<div> <div>All Documents</div> <div> Folders and Views  My Favorites  By Author  By Category  Archive Logs  Agents  Design </div> </div>		<div> <div>11/17/95</div> <div>SuperPro tennis racquets [Peter Carroll]</div> <div> Will this replace another line that we carry? [Susanna Levine]  Yes. NoPoint racquets are history [Peter Carroll 11/17]  These are big sellers and high quality [Stephanie Mahar 11/17]  Some good features about SuperPro [Micki Soler] 11/20  Jump-in-the-Boat fishing rods [Stephanie Mahar]  Sounds fishy to me [Michael Thompson] 12/12  Please get me more information [Stephanie Mahar 12/12]  More info about Jump-in-the-Boat [Stephanie Mahar 12/14]  Their numbers are wrong [David Yo 12/15] </div> </div>			
<div> <div>SuperPro tennis racquets</div> <div>Peter Carroll on 11/17/95 at 11:10 AM</div> <div>Category: New Products</div> </div>		<div> <div>W</div> <div>will soon be carrying SuperPro tennis racquets. A SuperPro</div> </div>			
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 2



**Figure 3 (PRIOR ART)**



### Figure 4

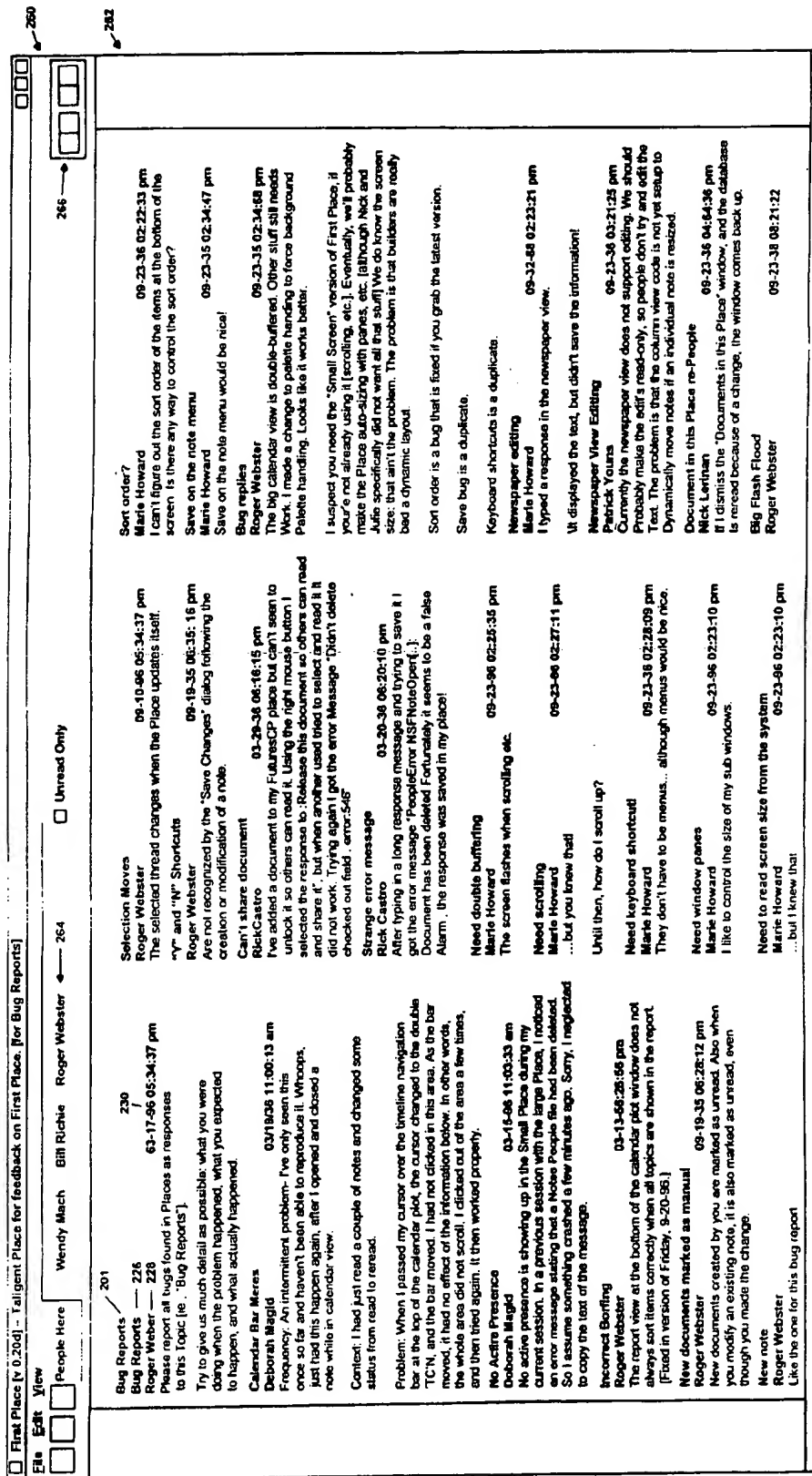
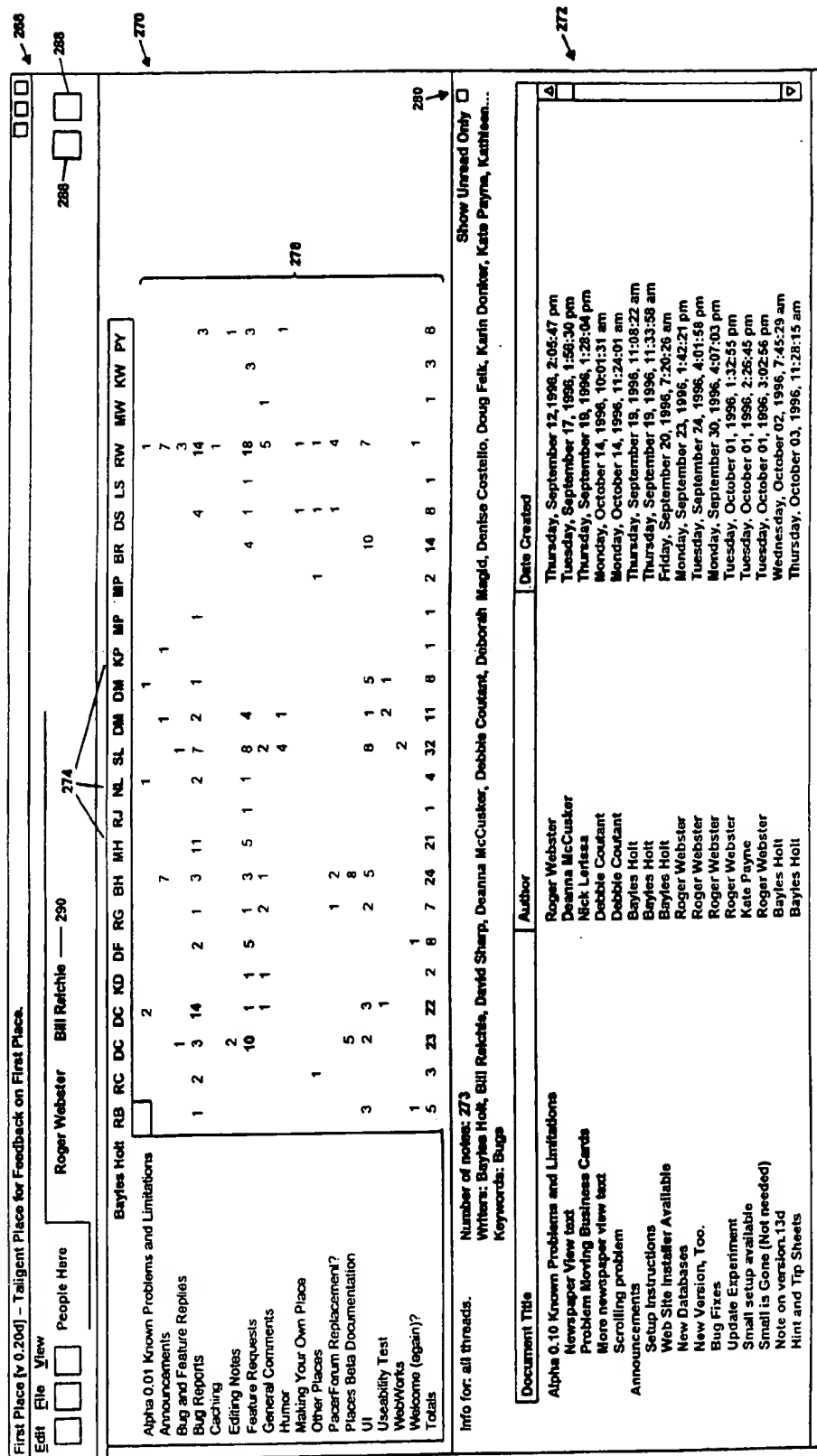
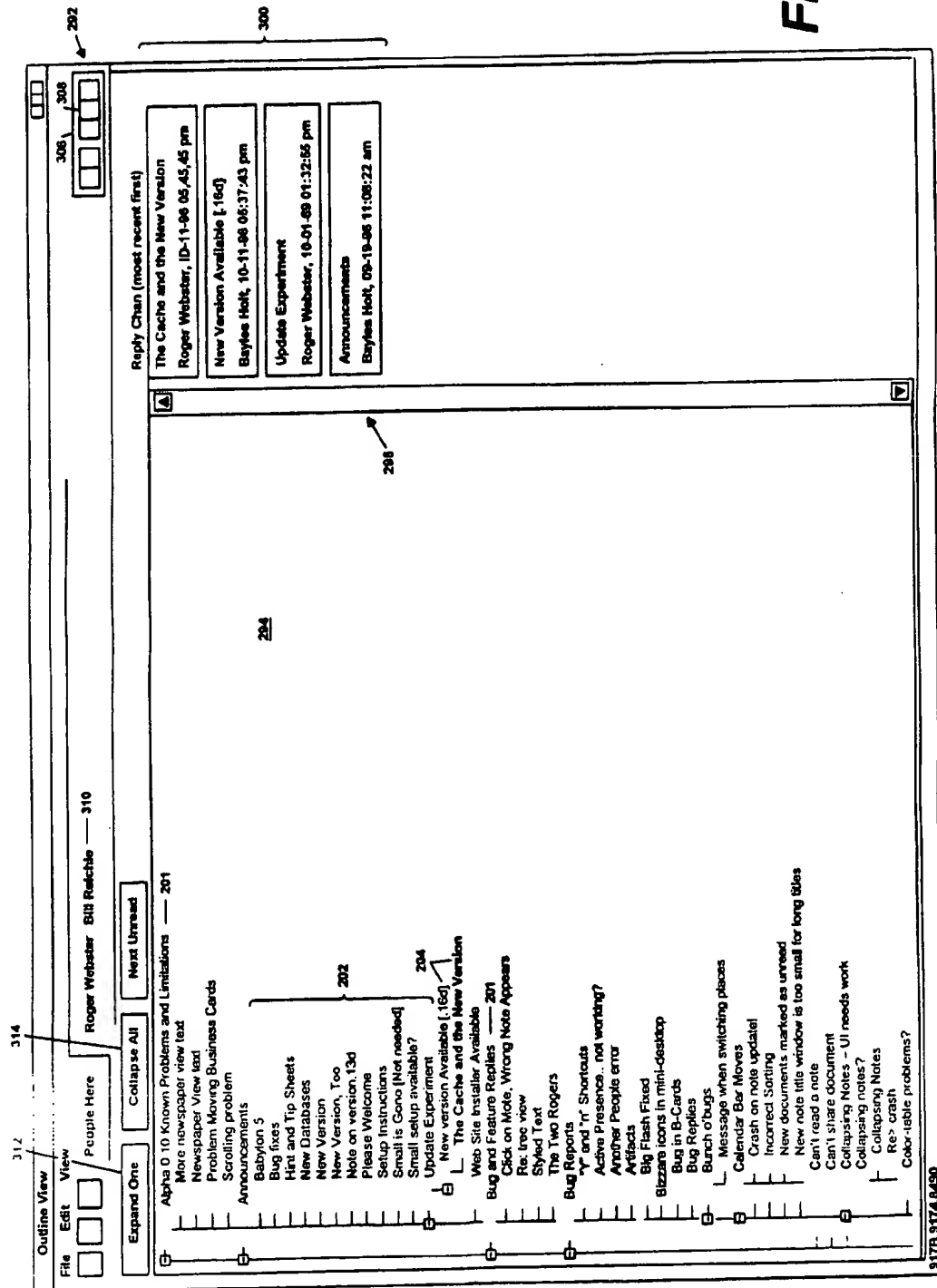


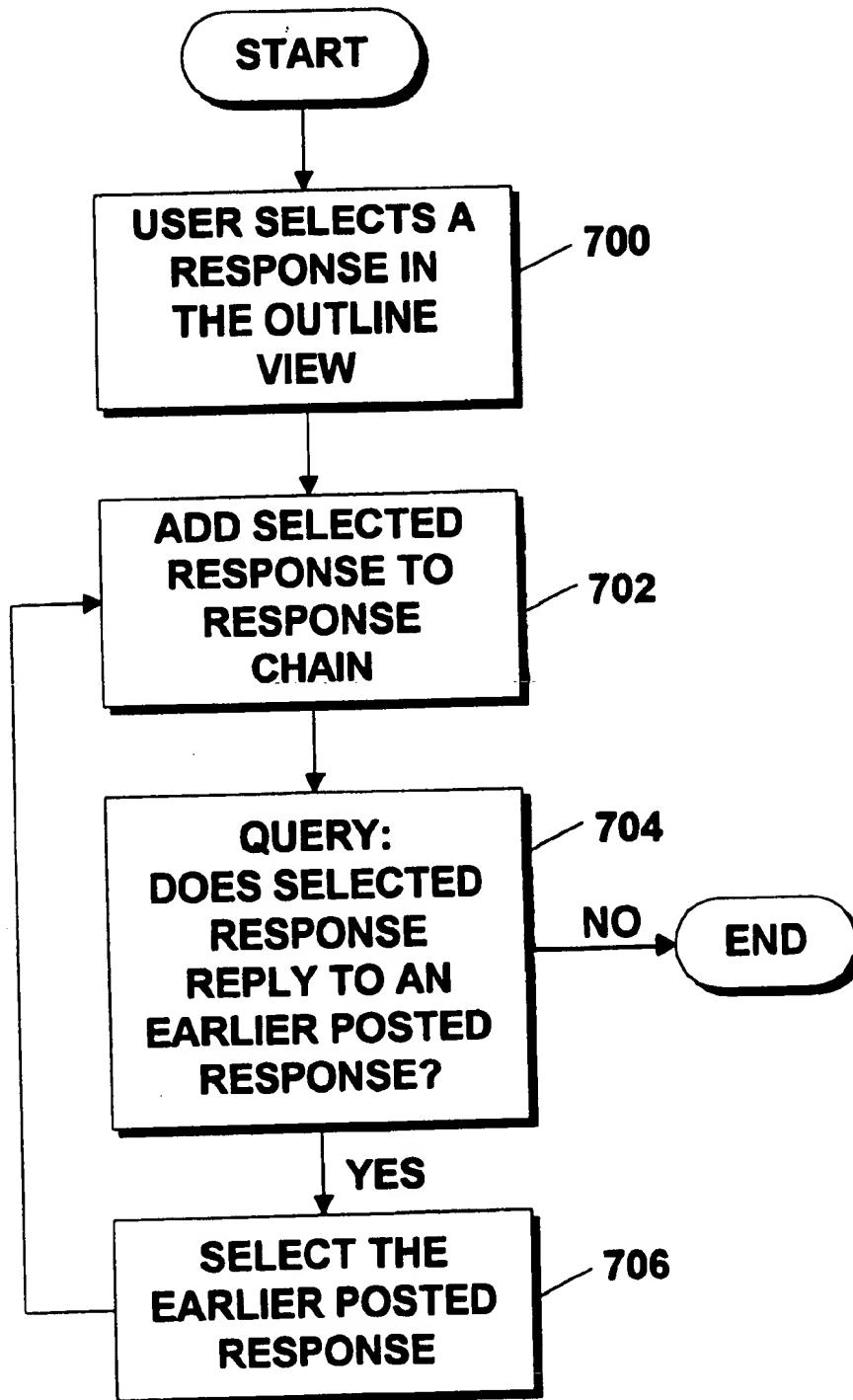
Figure 5



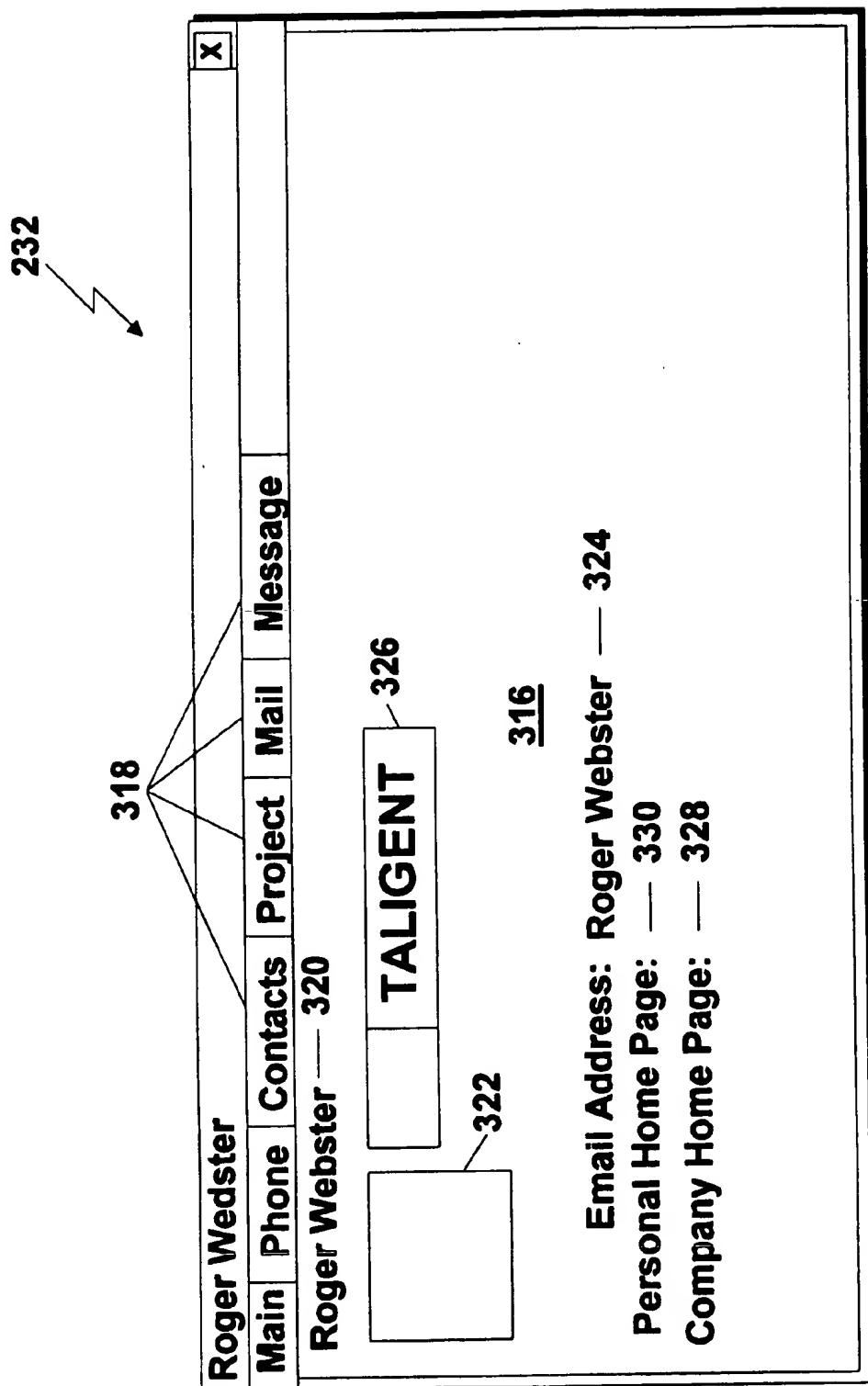
## Figure 6

Figure 7A





**Figure 7B**

**Figure 8**

## DATABASE GRAPHICAL USER INTERFACE WITH CALENDAR VIEW

### RELATED APPLICATIONS

This application contains subject matter which is related to the following applications: "Database Graphical User Interface with User Frequency View" (U.S. Ser. No. 08/777,616), "Database Graphical User Interface with Outline View" (U.S. Ser. No. 08/774,572) and "Database Graphical User Interface with Tabbed User View" (U.S. Ser. No. 08/774,576), all filed by Roger Webster and Nicholas Lerissa on an even date herewith and assigned to the present assignee.

### FIELD OF THE INVENTION

This invention relates generally to computer databases and, more particularly, to graphical user interfaces for accessing multiple-user databases.

### BACKGROUND OF THE INVENTION

Computer systems typically store large collections of data in databases in non-volatile computer memory. Data stored in databases commonly is organized into individual records which themselves are organized in a logical hierarchical manner that simplifies use of that database. Depending upon the database access policy, each record of the database may be accessed, reviewed, modified, or deleted by an authorized database user. When used in a business context for storing information relating to software development by a software development company, for example, data stored in the database may include data relating to projects being developed by the company, comments by employees relating to current technology of the company, and technical data related to ongoing company projects.

Most records in a database are accessible to a user through a "graphical user interface" (GUI). GUIs provide a visual metaphor of a real-world scene, such as a desktop, featuring icons that the user can manipulate with a pointing device, such as a mouse. The database user therefore may access a selected database by clicking on a GUI button representing the database, responsively causing the computer to display the contents of the selected database in an easy to understand format. This format itself is a GUI that enables the user to further access and manipulate the database.

One type of database application program product in widespread use is Lotus Notes™ (Notes), available from Lotus Development Corporation of Cambridge, Mass. Notes enables a user to construct a database that may be accessed by multiple users through a Notes-based GUI. FIGS. 1 and 2 show exemplary Notes GUIs used with a multi-user database that stores topics, responses replying to the topics (topic responses), and responses replying to the responses (reply responses). Notes organizes the data in the exemplary database by grouping the topics, and their corresponding responses, into folders.

The user may access the exemplary Notes database by clicking on a database access button (FIG. 1), which responsively displays the GUI shown in FIG. 2. The GUI in FIG. 2 includes a navigation pane for displaying the folders in the database, a view pane for viewing the topics and titles of the responses stored in the folders listed in the navigation pane, and a preview pane for displaying the content of a response selected in the view pane.

Although the Notes GUI displays some information about the topics and responses, such as the title of a response, date

of the response, and the author of the response, it does not tabulate a summary of this information and other related data. For example, summaries with indicia relating to the number of replies per user per topic are not displayed. Similarly, the Notes GUI does not display indicia or information relating to the number of replies to any one topic on each of several days, weeks, months, or years. Absent a GUI displaying such summaries, a user requiring this information necessarily would be required to manually tabulate these summaries while scrolling through the Notes GUI. Such a process is time consuming and inefficient.

Accordingly, it would be desirable to provide a GUI that summarizes database information in a simple, easy to understand, user-friendly format. It also would be desirable for such a GUI to facilitate user access to the database records directly from the GUI. It is among the general objects of the invention to provide such a device and method.

### SUMMARY OF THE INVENTION

In accordance with one aspect of the invention, a multiple user database may be accessed through a GUI which summarizes, in an easy and user-friendly format, the number of responses to any one topic on each of several days, weeks, months, years, or any other time designation. To that end, the GUI includes a calendar view and an adjacent report view. The calendar view is a two-dimensional grid having a timeline along a horizontal dimension, and an index listing a plurality of topics along the vertical dimension. The grid forms a plurality of intersecting horizontal topic rows and vertical time columns. Each intersection of a topic row and time column may include indicia that display the number of responses to the listed topics on the day, week, month, or year designated in a given time column. The report view includes a listing of the titles of each response to a topic selected in the index by a user.

In accordance with another aspect of the invention, the database may be accessed through a GUI which summarizes, in an easy and user-friendly format, the number of responses to a given topic by one or more of a plurality of users. To that end, the GUI includes a user frequency view and an adjacent report view. The user frequency includes a two-dimensional grid having indicia identifying one or more users along the horizontal dimension, and an index listing a plurality of topics along the vertical dimension. Similar to the previously described GUI, the grid forms a plurality of intersecting horizontal topic rows and vertical user identifying columns. Each intersection of a topic row and user identifying column may include indicia that display the number of responses to each of the listed topics by each of the users. The report view includes a listing of the titles of each response to a topic selected in the index, or the titles of each response by a user to a selected topic.

In accordance with yet another aspect of the invention, the database may be accessed by a GUI having an outline view that lists each of the responses under a topic in outline form, and an adjacent response chain view having indicia for accessing the content of each response in a response chain (described below) to a selected response. A response may either be a response to a topic (topic response) or a response to a previously posted response (reply response). The response chain therefore includes a selected response and any other responses that are directly linked, by responses, between the selected response and the topic.

In accordance with still another aspect of the invention, each of the above-mentioned inventive GUIs includes



graphical indicia indicating which users are simultaneously viewing the GUI. The indicia may be colored text.

In accordance with another aspect of the invention, the GUI displays information relating to a database user and includes means for connecting to the Internet and, in particular, to the user's world wide web home page. To that end, the GUI includes a display view having indicia related to the user, and a plurality of tabs adjacent to the display view for changing the indicia in the display view. Among the indicia in the display view is text identifying the world wide web home page address for the user. Clicking on the world wide web address causes the computer to directly connect to the user's home page.

It is among the objects of the invention to provide an apparatus and method that simplifies access to a database.

It is another object of the invention to provide an apparatus and method that summarizes database information in an easy, user-friendly format.

### BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects and advantages of the invention will be appreciated more fully from the following further description thereof with reference to the accompanying drawings wherein:

FIG. 1 is an illustration of a prior art Notes GUI for accessing a Notes database;

FIG. 2 is an illustration of a prior art Notes GUI displaying the contents of a Notes database;

FIG. 3 is a block diagram of a computer system suitable for use with the present invention;

FIG. 4 is an illustration of a first database GUI having a calendar view and a report view;

FIG. 5 is an illustration of a second database GUI having a response content view;

FIG. 6 is an illustration of a third database GUI having a user frequency view and a report view;

FIG. 7A is an illustration of a fourth database GUI having an outline view and a response chain view;

FIG. 7B is a flow chart illustrating the process of forming a response chain; and

FIG. 8 is an illustration of a fifth database GUI having a display view showing the world wide web home page address for a user.

### DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

FIG. 3 illustrates the system architecture for an exemplary computer system 100, such as an IBM PS/2® computer, on which the invention can be implemented. The exemplary computer system of FIG. 3 is for descriptive purposes only and should not be considered a limitation of the invention. Although the description below may refer to terms commonly used in describing particular computer systems, such as an IBM PS/2 computer, the described concepts apply equally to other systems, including systems having architectures that are dissimilar to that shown in FIG. 3.

The computer system 100 includes a central processing unit (CPU) 105, which may include a conventional microprocessor, random access memory (RAM) 110 for temporary storage of information, and read only memory (ROM) 115 for permanent storage of information. A memory controller 120 is provided for controlling system RAM 110. A bus controller 125 is provided for controlling bus 130, and an interrupt controller 135 is used for receiving

and processing various interrupt signals from the other system components.

Mass storage may be provided by diskette 142, CD-ROM 147, or hard disk 152. Data and software may be exchanged with computer system 100 via removable media, such as diskette 142 and CD-ROM 147. Diskette 142 is insertable into diskette drive 141, which is connected to bus 130 by controller 140. Similarly, CD-ROM 147 is insertable into CD-ROM drive 146, which is connected to bus 130 by controller 145. Finally, a hard disk 152 is part of a fixed disk drive 151, which is connected to bus 130 by controller 150.

User input to computer system 100 may be provided by a number of devices. For example, keyboard 156 and mouse 157 are connected to bus 130 by keyboard and mouse controller 155. An audio transducer 196, which may act as both a microphone and a speaker, is connected to bus 130 by audio controller 197. It should be obvious to those reasonably skilled in the art that other input devices such as a pen and/or tablet and a microphone for voice input may be connected to computer system 100 through bus 130 and an appropriate controller. DMA controller 160 is provided for performing direct memory access to system RAM 110. A visual display is generated by a video controller 165, which controls video display 170. Computer system 100 also includes a communications adapter 190 which allows the system to be interconnected to a network 195 via bus 191. The network 195 may be a local area network (LAN) or a wide area network (WAN).

Computer system 100 is generally controlled and coordinated by operating system software, such as the OS/2® operating system (available from the International Business Machines Corporation, Boca Raton, Fla.) or WINDOWS® NT (available from Microsoft Corp., Redmond, Wash.). Among other computer system control functions, the operating system controls allocation of system resources and performs tasks such as process scheduling, memory management, networking and I/O services.

FIG. 4 shows a first GUI 200 for displaying information stored in an exemplary Notes database. The exemplary Notes database stores a plurality of topics 201 and responses to those topics 201 by a plurality of users of that database. As shown in FIG. 4, such topics 201 may include, "Announcements," "Bug Reports," and "Humor." Users may post responses either directly to topics 201 (topic responses 202) or indirectly to topics 201 by replying to earlier posted responses (reply responses 204). For example, a topic response 202 entitled, "Calendar Bar Moves" was posted by Deborah Magid on Sep. 19, 1996 in response to the topic 201, "Bug Reports." Similarly, a reply response 204 was posted by Roger Webster on Sep. 20, 1996 entitled, "Incorrect Sorting." As shown in FIG. 5 (discussed in detail below), each response includes the title of the response, the author, the time the response was posted, and text conveying the substance of the response. Any user may post a response any number of times on a given day.

The database may be created according to known methods. The exemplary Notes database, for example, may be a relational database having a plurality of responses (i.e. records) with multiple fields. These fields may include:

- author of response;
- title of the response;
- topic being responded to;
- date and time of creation of response;
- title of earlier response being responded to, if any; and
- text written by author.

When creating a new record (i.e. response), the author enters the title of the response and the text. Notes automatically enters the remaining fields into the record.

The exemplary Notes database may be accessed simultaneously by a plurality of users. Accordingly, the first GUI 200 may be accessed and displayed on the display 170 by any of those users simultaneously. To that end, a second (non-relational) database is used by the invention to store information about such database users. This non-relational database may include a record type defined as "active presence," having a single field that stores an identification string identifying each of the users having the first GUI 200 displayed on their displays 170. A user's identification string is entered into the second database once the first GUI 200 is displayed on that user's display 170. That user's identification string is deleted from the second database once the first GUI 200 is not displayed on that user's display 170.

As noted above, the first GUI 200 (FIG. 4) displays information related to the exemplary database in an easy, user friendly format. Specifically, the first GUI 200 graphically displays the number of responses that were posted to each of the topics 201 on each day of a selected time period. For example, the first GUI 200 graphically indicates that the topic 201 "Bug Reports" had one response posted on Sep. 17, 1996, six responses posted on Sep. 19, 1996, two responses posted on Sep. 20, 1996, etc. . . . To effectuate this functionality, the first GUI 200 includes a calendar view 206 and an adjacent report view 208. The calendar view 206 is a two-dimensional grid having a timeline 210 along the horizontal dimension, an index 212 listing a plurality of topics 201 along the vertical dimension, and a plurality of horizontal markings 214. The horizontal markings 214 have one or more darkened rectangular boxes 216 aligned with the intersection of a topic 201 and a date. For example, the topic 201 "Bug and Feature Replies" has a first box lined up with Oct. 22, 1996, a second box lined up with Oct. 23, 1996, and a third darkened box lined up with Oct. 25, 1996. This alignment visually indicates that at least one response was posted to that topic 201 on those dates. No responses were posted to that topic 201 on any other days. The darkened boxes 216 also include lines 218 to visually indicate the number of responses on a given day. In the exemplary database, the number of responses is equal to one more than the number of lines 218 within each box so that the box is broken into a number of sub-boxes which equal the number of responses. For example, the topic 201 "Bug and Feature Replies" received one posting on Oct. 22 and 25 of 1996 (i.e. no lines 218 in the box 216), and three postings on Oct. 23, 1996 (two 218 in the box). Although rectangular boxes 216 with lines 218 are shown, any indicia may be used that effectuates the purpose of the first GUI 200.

The timeline 210 can be scrolled infinitely into the past and future. This is useful when a second, older database is combined with the exemplary database. When scrolling the timeline 210, responses that pass out of the field of view are graphically displayed as flags 213 that may be displayed either in a past space 215 or in a future space 217 on the calendar view 206. Clicking on one of the flags causes the computer to display a listing, in front of the calendar view 206, of the responses represented by the selected flag 213. The listing may include the author 228, title 226, and posting date 230 of each of the represented responses. Selection of one of the represented responses causes the timeline 210 to display the time period around the posting date of the selected represented response. Subsequent selection of a "today's date" button 219 returns the timeline 210 to the current date.

The calendar view 206 also may include two vertical lines 220 to visually block off a preselected block of time. As shown in FIG. 4, this block of time may be one month. Scroll buttons 222 may also be included to scroll the timeline 210.

Any one of the topics 201 listed along the vertical dimension of the calendar view 206 may be selected to display a list 224, in the report view 208, of each response posted to the selected topic 201. Accordingly, the report view 208 includes headings for listing the title 226, author 228, and creation date 230 of each response to the selected topic 201. When any one of the listed authors' names is clicked in the report view 208, a second GUI 232 having information about the selected author (FIG. 8, discussed below) will appear in front of the first GUI 200. The report view 208 also includes indicia 240 indicating the number of responses to the topic 201, the names of all authors who responded to the topic 201, and any keywords that may have been used. An unread only button 242 may also be included to display unread responses only. Response scroll buttons 244 may also be included to scroll through the responses listed in the report view 208.

The first GUI 200 further includes a listing of each user currently having the first GUI 200 displayed on their displays 170. This listing is displayed in an active presence view 248 on the first GUI 200 which begins with the text "People Here." The first GUI 200 obtains this information from the active presence records in the second database. The letters of the user's names may also be colored to indicate whether the first GUI 200 is in the foreground of the user's display 170. In the preferred embodiment, a user's name is displayed with red letters when the first GUI 200 is in the foreground, and black letters when the first GUI 200 is not in the foreground.

A means for displaying a summary of all responses posted on a specified date to a selected topic may be included in the first GUI 200. To that end, an enable button (not shown) may be included on the first GUI 200 that causes a response list window (not shown) to be displayed over the first GUI 200 when the cursor is passed over one of the darkened boxes 216 for a selected topic. The response list window may include the title 226 and author 228 of each response posted on that selected date. Double clicking on one of the responses listed in the response list window causes the user's computer system 100 to display the entire response over the first GUI 200.

The first GUI 200 may also include a plurality of buttons 252 to perform miscellaneous functions. For example, several buttons can be included to move to another GUI for that database, or to a GUI for another database. Similarly, buttons 254, 256 for creating a new topic 201 or a new response can also be included. A search button 258 may also be included to search for keywords in the titles of the responses.

During installation, one or more people may be assigned duties within the exemplary database. For example, one person may be assigned the duty of "administrator" to update and trouble-shoot problems with the exemplary database, and another person may be assigned the duty of "response reviewer" to review responses for content and relevance to particular projects. The duties are stored in a duty database, created upon installation of the exemplary database, which may include the fields "name" and "duty to be performed by named person." The duty database may be accessed by known means for review or modification by authorized users.

Each of the responses shown in the first GUI 200 may be directly accessed either by clicking on a response title listed

in the report view 208, or by clicking on one of the darkened rectangular boxes 216 on the calendar view 206. This causes a third GUI 260 to be displayed in front of the first GUI 200. As shown in FIG. 5, the third GUI 260 includes a response view 262 and an active presence view 264. The active presence view 264 shows the users displaying the third GUI 260. The function and operation of the third GUI active presence view 264 are identical to those of the active presence view 248 in the first GUI 200. The response view 262 includes the title, author, posting date, posting time, and content of each selected response(s). Preferably, the content of the selected responses cannot be edited unless the user viewing the third GUI 260 is the author of the posted response. The content of certain selected response may be omitted, however, to display the title, author posting date and posting time only. This is useful in a number of instances, such as when one or more of the responses had already been read by the user accessing the third GUI 260. A plurality of buttons 266 similar to those in the first GUI 200 may also be included to increase the functionality of this GUI.

A user may scroll through the responses displayed in the response view 262 in accordance with any method known in the art. It is preferred that the responses be scrolled in a "serpentine" manner in which each column scrolls into the adjacent column as the display is scrolled. For example, with reference to FIG. 5, when the left column (i.e. having the heading "Bug Reports") is scrolled upwardly by one response, the response at the top of the center column will be moved to the bottom of the left column. The response at the top of the right column consequently will move to the bottom of the center column. Each of the other responses will similarly move upwardly by one response. The response view 262 also may be scrolled on a column by column basis, or on a page by page basis.

The display of responses in the response view 262 may be customized to display selected groups of responses. To that end, means are included for displaying responses having common posting dates 230 for a selected topic, authors 228, and topics. For example, double-clicking on one of darkened boxes 216 (in the calendar view 206) will group according to responses to a selected topic on a specific date. As further examples, double-clicking on a date in the timeline 210 will group according to common posting dates 230, and double-clicking on an author's name in a fourth GUI 268 (discussed in detail below) will group according to one of the author's names.

FIG. 6 shows the fourth GUI 268 that summarizes, in an easy and user-friendly format, the number of responses to each of the topics 201 by each of the users. To that end, the fourth GUI 268 includes a user frequency view 270 and an adjacent report view 272. The user frequency view 272 is a two-dimensional grid having the initials 274 of each of the users along the horizontal dimension, and a listing 276 of the topics 201 along the vertical dimension. A plurality of numbers 278, each representing the number of responses to a topic 201 by a user, are positioned at the intersection of the initials 274 and the topics 201. For example, user "RB" posted one reply to the topic 201 "Bug Reports," three replies to topic 201 "UI," and one reply to topic 201 "Welcome (again)?". The numbers 278 can also have a colored background to visually indicate more or less usage by a user. In the preferred embodiment, the background successively changes from white (lowest), to yellow (low medium), then to orange (high medium), and finally to red (highest) as the number of responses increases.

Similar to the report view 208 in the first GUI 200, the report view 272 in the fourth GUI 268 lists the title of

selected responses, their authors, and the date and time that those responses were posted. In the preferred embodiment, those selected responses are selected by the user when one of the numbers 278 in the grid is clicked. This causes all of the responses represented by that number to be listed in the report view 272. For example, if the number "2" is double clicked under the initials 274 "DC," the two responses posted by user "DC" in reply to the topic 201 "Alpha 0.10 Known Problems and Limitations" will be listed in the report view 272.

The report view 272 in the fourth GUI 268 has a number of functions and features that are identical to those in the first GUI report view 208. A first such function is effectuated when an author's name is double-clicked, causing the second GUI 232 (FIG. 8) to be displayed in front of the fourth GUI 268, thus displaying information related to the selected author. A second function is effectuated when the title of the response is double-clicked, causing the third GUI 260 (FIG. 5) to appear showing the content of the response. An unread only button 280 that causes the report view 272 to show only unread responses also may be included. Further, like the first GUI report view 208, the number of responses, writers of the notes, and keywords may be displayed in the fourth GUI report view 272.

Like the first GUI 200, the fourth GUI 268 may also include a plurality of buttons (not shown) to perform miscellaneous functions. For example, several buttons can be included to move to another GUI. Similarly, buttons 286, 288 for creating a new topic or a new response may also be included. An active presence view 290 showing users displaying the fourth GUI 268 may also included. The function and operation of the fourth active presence view 290 are identical to those of the active presence view 248 in the first GUI 200.

FIG. 7A shows a fifth GUI 292 having an outline view 294 that lists each of the responses in the database under the topic 201 to which they respond. Specifically, the topics 201 listed in the outline view 294 are not indented while topic responses 202 are indented one unit to the right more than the topics 201. Similarly, reply responses 204 are grouped under the previously posted responses to which they respond (parent responses) and are indented one unit to the right more than their respective parent responses. This format enables a user to quickly and easily differentiate between topic responses 202 and reply responses 204. It also visually indicates the relationship between different responses in the database.

The fifth GUI 292 also has a button view 296 having buttons 300 for accessing each response in a selected response chain. FIG. 7B details the process of forming a response chain. Specifically, a response is selected by a user in the outline view 294 at step 700. This selected response is added to the response chain at step 702. If the selected response is not a reply to an earlier posted response, the process is ended, thus yielding the fully formed response chain (step 704). If the selected response is a reply to an earlier posted response, that earlier selected response is selected at step 706 and fed back to step 702. This process continues until the response chain is fully formed. After the response chain is fully formed, a button 300 for each response in the response chain are formed in the button view 296. Clicking on one of the buttons 300 causes the computer to display the response whose title is displayed on the button 300.

Like the first GUI 200, the fifth GUI 292 may also include a plurality of other buttons to perform miscellaneous functions. For example, several buttons (not shown) can be

included to move to another GUI. Similarly, buttons 306, 308, and 309 for creating a new topic 201, a new topic response 202, or a new reply response 204 may also be included. An active presence view 310 showing users displaying the fifth GUI 292 may also be included. The function and operation of the fifth active presence view 310 are identical to those in the first GUI 200. The fifth GUI 292 may also have expand and collapse buttons 312, 314 to expand or collapse the listing of responses in the outline view 294.

As previously noted, the second GUI 232 (FIG. 8) is displayed when an author's name is double clicked in either the first GUI report view 208 or the fourth GUI report view 272. The second GUI 232 both displays information related to the selected author, and enables the selecting user to send an electronic message to the selected author. To that end, the second GUI 232 includes a display view 316 and a plurality of labeled tabs 318 adjacent to the display view 316. When a tab is clicked, the display view 316 displays indicia that is related to the clicked tab label. For example, as shown in FIG. 8, the preferred embodiment includes the tabs 318 "main," "phone," "contacts," "project," "mail," and "message." Each of these tabs 318 provides indicia related to the labeled topic 201. The indicia may be buttons or other indicia that have a preselected function when clicked.

In the preferred embodiment (FIG. 8), when the "main" tab is selected, the computer displays the author's name 324, picture 322, email address 324, company logo 326, company world wide web home page address 328, and personal world wide web home page address 330. This information is stored in a Notes-based global address book database which is created when the Notes system is installed. The address book database includes information for display by each of the tabs. Clicking on either of the world wide web home page addresses 328, 330 causes the computer to connect directly to the selected world wide web home page. This connection is made through conventional means. The "phone," "contacts," and "project" tabs 318 respectively include indicia related to the author's various business and personal phone numbers (and fax numbers), relatives of the author to contact in an emergency, and company projects involving the author. The "mail" tab displays the electronic mail in the author's electronic mailbox queue.

The second GUI 232 may be customized by known methods, such as a Wizard program that guides a user through setting and/or modifying the display of that user's data in the second GUI. The user may therefore add and/or move images, logos, personal or professional data, and other information as desired.

An electronic message may be sent to the author by selecting the "message" tab. This causes the computer to display an electronic mail template having a message space for the user to enter a text message. After entering a text message, the user sends the message to the author by conventional means. Since both the source and destination of the electronic mail message are known, both are automatically entered into the message for transmission to the author. After the message is sent, it is temporarily stored in the second database as a data type "message" having fields "to," "from," and "body of message." The next time either of the first, second, third, fourth, or fifth GUIs are displayed by the author's computer, the message will be transmitted to and displayed by the author's display 170. The message record is deleted from the second database after receipt of the message by the author. If any one of the inventive GUIs are displayed by the computer when the message is sent, it is expected that the message will be displayed by such author's computer within two to three seconds of transmission.

The message tab also includes means for sending instant messages to an author's display 170. After being sent, instant messages are displayed immediately (i.e., within two to three seconds) on the receiving author's display 170. In the event that the receiving author is not accessing the exemplary database, the instant message will be displayed immediately after the receiving author subsequently accesses the exemplary database. Like the previously discussed electronic messages, after selecting the message tab, the computer displays an electronic mail template having a message space for the user to enter a text message. Since both the source and destination of the instant message are known, both are automatically entered into the message for transmission to the selected author. After entering the text message and selecting a "send instant message" command button, the text message is displayed on the receiving author's display immediately, or immediately after receiving author's subsequent access into the exemplary database. After selecting the send instant message command button, the instant message is temporarily stored in the second database, until deleted by the receiving author, as a data type "message" having fields "to," "from," and "body of message." Unlike the previously discussed electronic messages, an instant message does not require a receiving author to check an electronic mailbox for messages.

Instant messages may also be sent to any number of users currently using the exemplary database. To that end, the electronic mail template may include an "additional users" space for directing the instant message to additional users of the exemplary database. User's names may be entered into the additional users space to direct the instant message to the preselected additional users. Names entered into the additional users space are automatically entered into the field "to" in the second database.

Each database having topics, topic responses 202, and reply responses 204 has a set of the previously discussed inventive GUIs for their access. Although a Notes-based database was discussed in connection with the invention, any known database application program product may be used in connection with the inventive GUIs. The database may reside on a server in a local area network, or on a remote storage site that is accessible through the Internet.

The invention may be used with many different types of databases having different proprietary formats by means of a database interface module. Such types of databases may be accessed through the Internet. Among other advantages, the interface module enables the invention to be updated without requiring the updating code to be directed to every different type of database that may be accessed. The interface module may be considered to be a means for iterating through both file system entities (e.g., volumes, directories, and files) and database records from a wide range of different proprietary databases. To that end, the interface module includes a code library that is implemented as a dynamic link library ("DLL"). The DLL creates a two-dimensional table, from the information in the databases being accessed, that is in a form readable to the invention. The rows of the two-dimensional table are either database records or file system entities, and the columns are, respectively, either database fields or file system attributes. The table is formed from input data that may be either a PDS file for database records and file system entities, or a path to a file system directory for file system entities only. A PDS file is a file (with a .pds extension) containing information that allows the DLL to establish a data source (i.e. the two-dimensional table) for either file system entities or database records. The contents of a PDS file are known in the art.

The protocol for establishing a data source (i.e. the two-dimensional table) from a data source specification is as follows:

- OpenDataSource: takes a fully-qualified path to either a PDS file or a directory, and returns a "handle" for use in subsequent conversations with the library;
- CloseDataSource: takes a handle to a previously opened data source and frees all resources associated with the Data Source;
- FirstRow: takes a handle to a previously opened data source; moves the internal table row index to the beginning of the data source; returns true if successful;
- NextRow: takes a handle to a previously opened data source; moves the internal table row index to the next logical record in the data source; returns true if successful;
- Columns: takes a handle to a previously opened data source; returns the number of columns available;
- Rows: takes a handle to a previously opened data source; returns -1 for database data sources (since the total number of records is often not known until the database has been completely iterated through), and the actual number of rows in a file system data source;
- Column: takes a handle to a previously opened data source, the index of the column of interest, and a pointer to a string to be filled in with the result;
- ColumnType: takes a handle to a previously opened data source and the index of the column of interest; returns a type code indicating the actual type of the data in the specified column;
- DisplayColumnName: takes a handle to a previously opened data source and the index of the column of interest; in the case of a database data source, a configuration-time supplied user-friendly name field is returned (this info is stored in the PDS). In the case of a file system data source, the field names are TBD.
- ActualColumnName: takes a handle to a previously opened data source and the index of the column of interest; in the case of a database data source, the actual name of the database field is returned (this information is stored in the PDS). In the case of a file system data source, this call returns the same information as DisplayColumnName.

In an alternative embodiment, the interface module also may be, for example, the record maintenance interface disclosed in U.S. Pat. No. 5,522,066 (Liu), the disclosure of which being incorporated herein by reference, in its entirety.

In another alternative embodiment, the invention may be implemented as a computer program product for use with a computer system. Such implementation may include a series of computer instructions fixed either on a tangible medium, such as a computer readable media (e.g. diskette 142, CD-ROM 147, ROM 115, or fixed disk 152 as shown in FIG. 3) or transmittable to a computer system, via a modem or other interface device, such as communications adapter 190 connected to the network 195 over a medium 191. Medium 191 may be either a tangible medium (e.g., optical or analog communications lines) or a medium implemented with wireless techniques (e.g., microwave, infrared or other transmission techniques). The series of computer instructions embodies all or part of the functionality previously described herein with respect to the invention. Those skilled in the art should appreciate that such computer instructions can be written in a number of programming languages for use with many computer architectures or operating systems.

Furthermore, such instructions may be stored in any memory device, such as semiconductor, magnetic, optical or other memory devices, and may be transmitted using any communications technology, such as optical, infrared, microwave, or other transmission technologies. It is expected that such a computer program product may be distributed as a removable media with accompanying printed or electronic documentation (e.g., shrink wrapped software), preloaded with a computer system (e.g., on system ROM or fixed disk), or distributed from a server or electronic bulletin board over a network (e.g., the Internet or World Wide Web).

Each of the inventive GUIs may be constructed by conventional software programming techniques known in the art. It is preferred that the GUIs be constructed by visual builders, such as OCX™ (available from Microsoft Corp.) or Delphi™ (available from Borland Corp.). Database functions, such as storing information, searching through records, accessing records, and modifying records, are all performed by the underlying database application program product. The exemplary database uses Notes to perform each of those functions.

Although various exemplary embodiments of the invention have been disclosed, it will be apparent to those skill in the art that various changes and modifications can be made which will achieve some of the advantages of the invention without departing from the spirit and scope of the invention. These and other obvious modifications are intended to be covered by the appended claims.

Having thus described the invention, what we desire to claim and secure by Letters Patent is:

1. Apparatus for use with a computer system having a memory and a display, the apparatus residing in the memory and generating a graphical user interface displaying information relating to a database storing a plurality of topics and responses to those topics, each response having a content, title, author, and posting date, the graphical user interface comprising:

a calendar view having a two-dimensional grid in which a timeline is displayed along one dimension of the grid, and an index listing the plurality of topics is displayed along the other dimension of the grid, the timeline displaying a plurality of equal time units, the calendar view further having means for graphically displaying the number of responses posted to each topic during the time designated by each time unit; and

a report view adjacent to the calendar view, the report view having means for displaying indicia relating to each response posted to one of the plurality of topics.

2. The apparatus as defined by claim 1 wherein the calendar view includes means for causing the content of each response posted to the one of the plurality of topics to be displayed in front of the graphical user interface.

3. The apparatus as defined by claim 1 further including means for displaying the content of the response in the form of a column.

4. The apparatus as defined by claim 3, wherein the displaying means displays the content of a plurality of responses in two columns.

5. The apparatus as defined by claim 4, further including means for scrolling the responses in a serpentine manner.

6. The apparatus as defined by claim 3, wherein the displaying means includes means for displaying preselected groups of responses.

7. The apparatus as defined by claim 1 wherein the report view includes means for displaying the title, author, and posting date of each response posted to the one of the plurality of topics.

## 13

8. The apparatus as defined by claim 1 wherein the response has an author, the report view including:

means for accessing a second database having information related to the author; and

means for displaying the author information in front of the graphical user interface.

9. The apparatus as defined by claim 1 wherein the first graphical user interface includes means for identifying one or more users simultaneously using the graphical user interface.

10. The apparatus as defined by claim 1 wherein the report view includes means for displaying the contents of the response to the topic in front of the graphical user interface.

11. The apparatus as defined by claim 1 wherein the report view includes means for excluding indicia from being displayed in the report view, the excluded indicia begin indicia identifying previously read responses.

12. The apparatus as defined by claim 1, wherein a duty database is stored in the memory, the duty database containing information related to duties of selected users of the database.

13. The apparatus as defined by claim 1, wherein the calendar view has a field view, the apparatus further including (1) means for scrolling the timeline; and (2) means for displaying responses not in the field of view.

14. The apparatus of claim 13, wherein the calendar view has a current field of view, the apparatus further including means for directly returning to the current field of view.

15. A method of summarizing information stored in a database, the method being for use with a computer system having a memory and a display, and generating a graphical user interface displaying information related to a database on the display, the method comprising the steps of:

A. accessing the database, the database containing a plurality of topics and responses posted to those topics, each response having a content, title, author, and posting date;

B. displaying a two-dimensional grid on a display device, the grid having a timeline designating a plurality of equal time units along a first dimension, a list of the plurality of topics stored in the database along a second dimension, and grid indicia aligned with both one of the equal time units and one of the topics, the grid indicia indicating that a response was posted to one of the of topics at a time designated by the one of the equal time units; and

C. displaying adjacent to the grid, in response to a user selecting one of the plurality of topics in the second dimension of the grid, report indicia related to each response posted to the selected topic.

16. The method of summarizing data in a database as defined by claim 15 wherein the report indicia includes the title, author, and posting time of each response posted to the selected topic.

17. The method of summarizing data in a database as defined by claim 15 comprising the further step of:

D. displaying indicia indicating the total number of responses posted to the one of the topics on the one of the equal time units.

18. The method of summarizing data in a database as defined by claim 15 comprising the further step of:

## 14

E. displaying, in response to a user selecting the grid indicia, the content of the response posted to the one of the plurality of topics at the time designated by the one of the equal time units.

19. The method of summarizing data in a database as defined by claim 15 comprising the further step of:

F. displaying, in response to a user selecting the report indicia, the content of the response related to the selected report indicia.

20. The method of summarizing data in a database as defined by claim 15 further comprising the steps of:

G. accessing a second database containing the names of the users displaying the two-dimensional grid; and

H. displaying the names of the users listed in the second database.

21. A computer program product for use with a graphics display device, the computer program product generating a graphical user interface on the display device for summarizing information stored in a database containing a plurality of topics and responses posted to those topics, each response having a content, title, author, and posting date, the computer program product comprising a computer usable medium having computer readable program code thereon, the computer readable program code including:

program code for accessing the database;

program code for displaying a two-dimensional grid on a display device, the grid having a timeline designating a plurality of equal time units along a first dimension, a list of the plurality of topics stored in the database along a second dimension, and grid indicia aligned with both one of the equal time units and one of the topics, the grid indicia indicating that a response was posted to one of the of topics at a time designated by the one of the equal time units; and

program code for displaying adjacent to the grid, in response to a user selecting one of the plurality of topics in the second dimension of the grid, report indicia related to each response posted to the selected topic.

22. The computer program product as defined by claim 21 further including program code for displaying indicia indicating the total number of responses posted to the one of the topics on the one of the equal time units.

23. The computer program product as defined by claim 21 further including program code for displaying, in response to a user selecting the grid indicia, the content of the response posted to the one of the plurality of topics at the time designated.

24. The computer program product as defined by claim 21 further including program code for displaying, in response to a user selecting the report indicia, the content of the response related to the selected report indicia.

25. The computer program product as defined by claim 21 further including:

program code for accessing a second database containing the names of the users displaying the two-dimensional grid; and

program code for displaying the names of the users listed in the second database.

\* \* \* \* \*